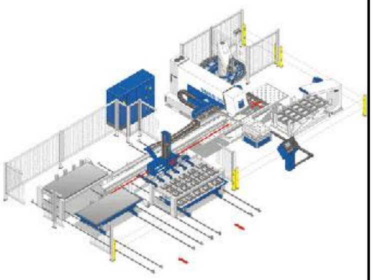
Université Internationale de Casablanca

Automatisme Logique



Pr. Khalid BENJELLOUN
bkhalid@emi.ac.ma
benjkhalid@gmail.com

Département Electrique
Section Automatique et Informatique Industrielle
Université Mohammed V –Agdal
ECOLE MOHAMMADIA D'INGÉNIEURS

P L A N

Chapitre 1: Introduction à l'automatisme logique (2h)

Chapitre 2: La logique combinatoire (6hs+4hs_TD)

Chapitre 3: La logique séquentielle(6hs+4hs_TD)

Chapitre 4: Grafcet (4hs+4hs_TD)

1. Introduction à l'automatisme logique

1. Introduction à l'automatisme logique

1.1 : Définitions

- Selon les techniciens :

  « L'automatisation consiste à *rendre automatique* les opérations qui exigeaient auparavant l'intervention humaine » *Encyclopédia Universalis*

- Une autre définition :

  « L'automatisation est considérée comme l'étape d 'un progrès technique où apparaissent des dispositifs techniques susceptibles de seconder l'homme, non seulement dans ses efforts musculaires, mais également dans son travail intellectuel de surveillance et de contrôle. » *Encyclopédia Universalis*

1. Introduction à l'automatisme logique

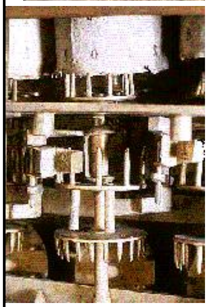
1.2 : Historique – Aujourd'hui

- Depuis les années 60 les ordinateurs sont en pleine expansion et sont intégrés à part entière dans tous les processus d'une entreprise.
- Avec la lutte féroce qui ce joue, les entreprises ne doivent pas seulement optimiser les équipements, mais aussi la façon d'intégrer le marché et la façon de gérer leur entreprise.
- Tous ces aspects vous seront montrés tout au long de ce cours!

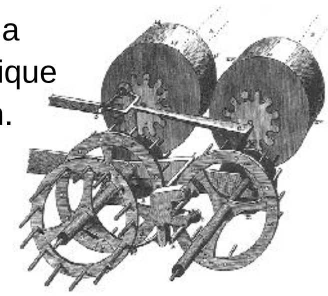
Historique – Les précurseurs

- **Blaise Pascal (1623—1662) :**
  Automatisation du calcul
  (*La pascaline)*

Historique – Les précurseurs (suite)



- **Afin d'aider son père dans son travail d'administration fiscal Blaise Pascal invente une machine à additionner et soustraire.**
  - Pour y arriver il a dû utiliser le principe de représentation des nombres en binaire! C'est la première fois que l'on applique ce genre de représentation.

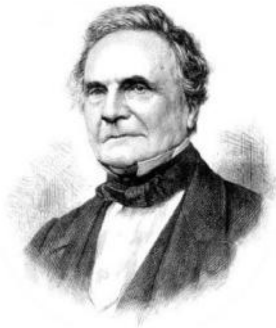


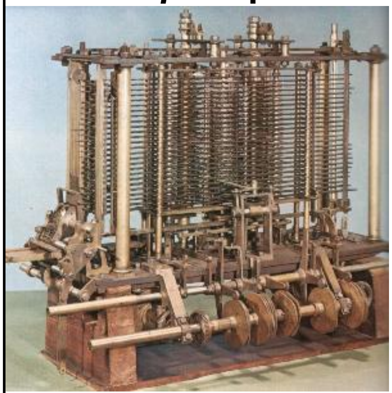
Historique – Les précurseurs (suite)

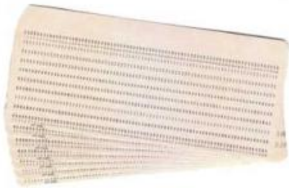
- **On doit aussi à Blaise Pascal quelques découvertes importantes appliquées encore aujourd'hui en génie :**
  - ***Pression atmosphérique*** : Étude sur le vide produit dans une colonne de mercure.
  - ***Calcul différentiel et intégral***: Étude sur les cycloïdes et les volumes de révolution.

Historique – Les précurseurs (suite)

- **Charles Babbage (1792—1871)**
  Programmation des métiers à tisser *Jacquard* par carte perforée.







Historique – Les précurseurs (suite)

- **Les travaux de Babbage sont à l'origines de l'invention de l'ordinateur.**
- **Il inventa le principe de la carte perforée qui sera utilisé au moins jusqu'à la fin des années 1970.**
- **Afin de programmer les métiers à tisser il pensa à un calculateur universel possédant :**
  - *système de gestion entrées/sorties ;*
  - *mémorisation interne ;*
  - *transfert de données ;*
  - *organe de commandes ;*
  - *opérateur arithmétique.*
- **Malheureusement la machine n'a jamais été implantée à cause de la technologie rudimentaire de l'époque.**

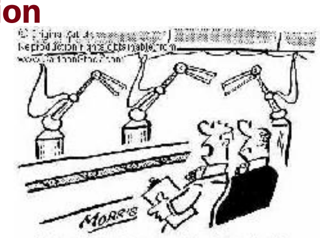
1. Introduction à l'automatisme logique

1.3 : Buts de l'automatisation

- Éliminer les tâches répétitives ou sans intérêt
  (ex: lavage du linge ou de la vaisselle…)
- Simplifier le travail de l'humain
  (Toute une séquence d'opération remplacée par l'appui sur un poussoir)
- Augmenter la sécurité
  (Éviter les catastrophes)
- Accroître la productivité
  (cadences de production plus élevées, pas de fatigue)
- Économiser les matières premières et l'énergie
  (production plus efficace)
- Maintenir la qualité

1. Introduction à l'automatisme logique

1.4 : Conséquences de l'automatisation



"The robots have gone on strike - they want parity with the robots at Ford."

- Augmentation du taux de production
- Diminution du coût d'achat des produits
- Uniformité dans les produits manufacturés
- Réduction des accidents de travail
- Opérations hasardeuses possibles
- Diminution des emplois…
- On remarque une diminution de la main d'œuvre par unité produite.
- Diminution des emplois pour travailleurs non qualifiés et augmentation des emplois pour les travailleurs qualifiés
- Certains types d'emplois deviennent très monotones et répétitifs
  (ex: inspection et surveillance des machines

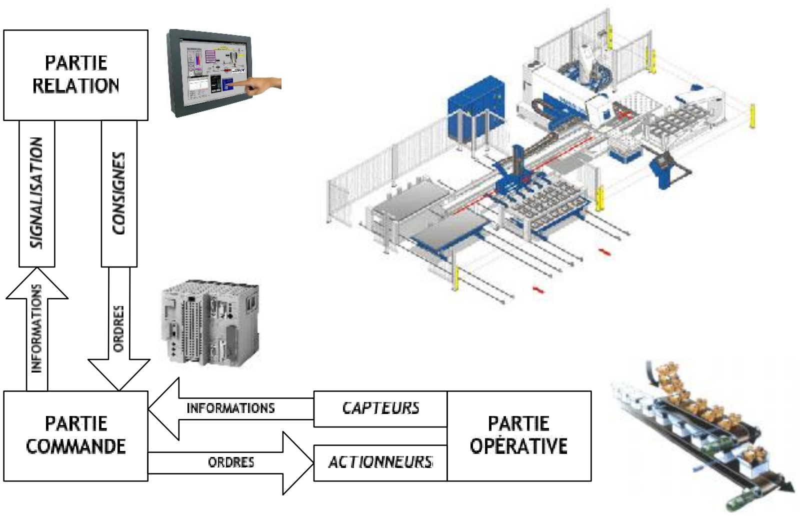
1. Introduction à l'automatisme logique

1.5 : Introduction à la logique

- **La logique oui ! Mais pour les automatismes!**
- **Les automatismes industriels connaissent une expansion importante, à cause:**
  - progrès considérable de la technologie,
  - développement des mécanismes de production qui exigent un plus haut niveau d 'automatisation pour des raisons de sécurité, de qualité, de compétitivité,
  - Avènement des systèmes informatiques à base de microprocesseurs.
- **Deux classifications des circuits logiques peuvent être envisagées**
  - la première, à caractère fonctionnelle, permet de distinguer 2 types de circuits logiques:
    - ceux pour lesquels la notion de temps n 'intervient pas, ce sont les circuits combinatoires
    - ceux pour lesquels la notion de temps intervient, ce sont les circuits séquentiels
  - la deuxième, à caractère technologique, permet de distinguer:
    - les circuits logiques câblés et dont la caractéristique demeure la rapidité
    - les circuits logiques programmés dont la caractéristique principale est l 'évolutivité

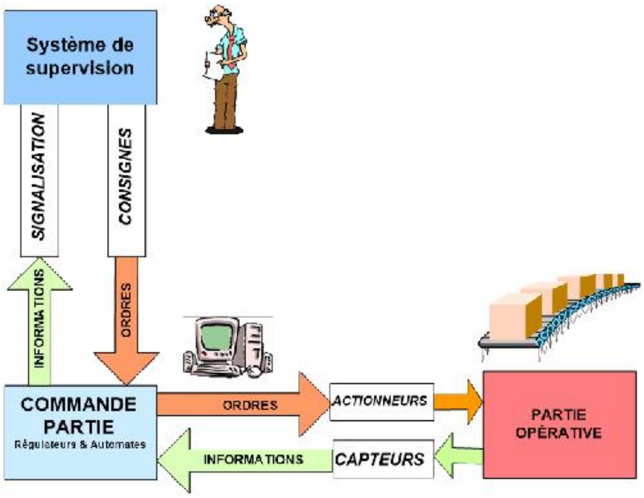
1. Introduction à l'automatisme logique

1.6 : Architecture de contrôle d'un système

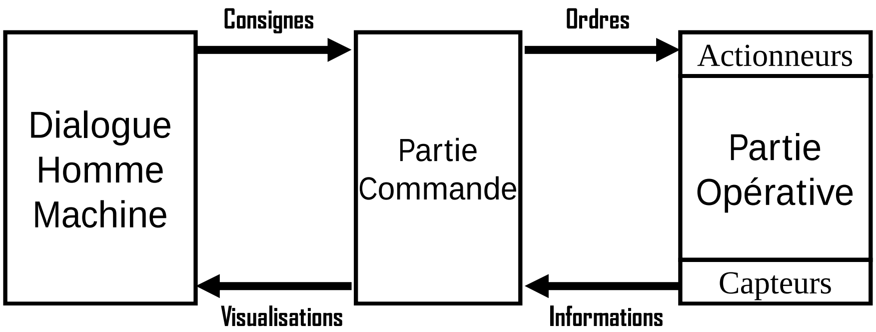


1. Introduction à l'automatisme logique

1.6 : Architecture de contrôle d'un système



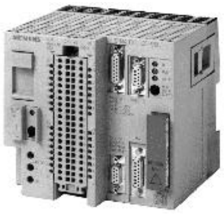
1. Introduction à l'automatisme logique



à concevoir

Structure d'un automatisme

- La partie commande
  - Automates programmables
  - Séquenceurs (électromécaniques ou pneumatiques)
  - Microcontrôleurs
  - Cartes dédiées
  - Etc.



Structure d'un automatisme (suite)

- La partie opérative
  - Moteurs électriques (CA ou CC)
  - Vérins (pneumatiques ou hydrauliques)
  - Vannes (électriques ou pneumatiques)
  - Éléments chauffants
  - Etc.

Structure d’un automatisme (suite)

- La partie relation
  - Panneaux de commande
    - Voyants, indicateurs
    - Poussoirs, sélecteurs
  - Interfaces Homme-Machine
  - Alarmes
  - Etc.

Structure d’un automatisme (suite)

- Ces trois parties comprennent :
  - Des fonctions ou organes binaires ;
  - Des fonctions de logique combinatoire ;
  - Des fonctions de logique séquentielle.

1. Introduction à l'automatisme logique

1.7: Signal numérique

Un signal numérique n'existe qu'à des niveaux ou états bien spécifiques. Il change de niveau ou d'état seulement selon des pas discrets.

La plupart des signaux numériques ont deux états.

Un système a deux états permet l'application de la logique booléenne ainsi que la représentation des nombres binaires qui sont le fondement de la conception de tous les circuits numériques.

Un signal numérique est traité par des circuits numériques classés selon leurs fonctions en :

- logique combinatoire : la sortie dépend de la valeur instantanée de l'entrée
- logique séquentielle: l'historique de la séquence dans le temps joue un rôle important dans la détermination de la sortie.

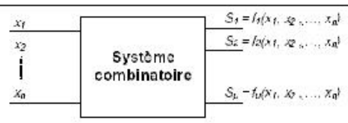
1. Introduction à l'automatisme logique

1.7.1 : Les systèmes combinatoire

– Définition : *L'état logique des sorties est fonction de l'état des entrées*

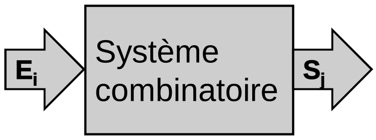
– Applications :
  - Circuits de sécurité et de verrouillage
  - Systèmes séquentiels simples

– Méthode de résolution :
  - Tables de Karnaugh ou de Mahoney



Système combinatoire générant p sorties S_i à partir des n variables d'entrée x

Logique combinatoire



- **Plan**
  - Algèbre de Boole
  - Représentation des fonctions logiques
    - Formes technologiques
    - Logigrammes
    - Chronogrammes
  - Simplification des fonctions logiques
  - Circuits combinatoires

1. Introduction à l'automatisme logique

1.7.2 : Les systèmes séquentiels

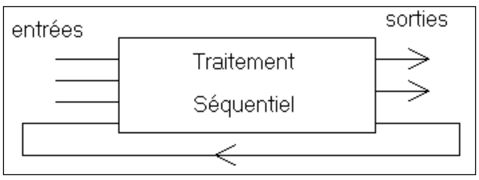
– Définition : *L'état logique des sorties est fonction de l'état des entrées et du passé du système*

– Applications :

- Toutes tâches de nature séquentielle, ascenseur, démarrage d'une centrale électrique

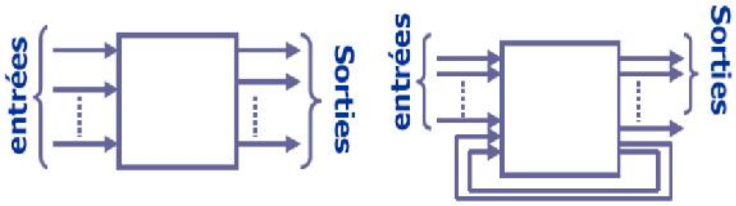
– Méthode de résolution :

- Méthode basée sur la logique combinatoire, Huffman
- Méthodes intuitives (géométriques)
- GRAFCET



1. Introduction à l'automatisme logique

L'état logique des sorties d'un système séquentiel est fonction de l'état des entrées et du passé système (système à mémoire), Alors que les sorties d'un système de logique combinatoire ne sont fonctions que des entrées à un instant donné,



1. Introduction à l'automatisme logique

1.8 : Outils utilisés

- **Algèbre de Boole**
- **Table de Karnaugh**
- **Méthode d'Huffman**
- **Grafcet**
- **...**

l 'objectif étant la modélisation, la simplification et la commodité de réalisation

1. Introduction à l'automatisme logique

1.9 : Situer le cours

- **Logique combinatoire**
- **Réalisations câblées de logique combinatoire**
- **Exposé et réalisation de logique séquentielle**
- **Le Grafcet, outil de spécification et d'automatisation**
- **Étude et application des règles du Grafcet**

2. La logique combinatoire

2. La logique combinatoire

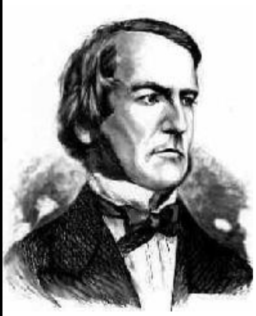
2.1 Introduction aux fonctions logiques

- **Systèmes binaires**
  - Deux états fondamentaux et distincts;
  - Vrai/Faux, Marche/Arrêt, Oui/Non.

- **Par convention:**
  - Un état est représenté par « 0 »;
  - L'autre est représenté par « 1 ».

2. La logique combinatoire

2.2 La logique Booléenne



- **En 1847, George Boole invente une algèbre pour traiter les variables binaires.**
  - Il écrira « The Mathematical Analysis of Logic », Cambridge,

- **Il définit 3 opérateurs de base, ainsi qu'une foule de règles et de postulats.**

NON

ET

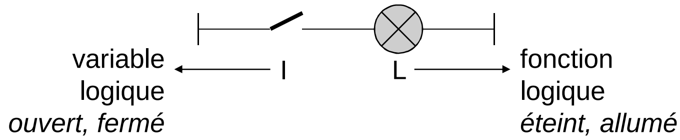
OU

Automatisme Logique- 31

Algèbre de Boole (binaire)

- **Définitions**
  - État logique (binaire ou discret)
    - Élément nul : valeur binaire **0** (faux, non, bas, ouvert, éteint, vide)
    - Élément unité : valeur binaire **1** (vrai, oui, haut, fermé, allumé, plein)
  - Variable logique (bit : **bi**nary dig**it**)
    - Grandeur représentée par un symbole (lettre ou signe) qui peut prendre 2 états logiques dans le cadre de l'algèbre de Boole.
  - Fonction logique
    - Fonction représentée par des groupes de variables réliés par des opérateurs logiques qui ne peut prendre que 2 états logiques **0** (point faux) ou **1** (point vrai).

système binaire :



Pr. K. BENJELLOUN | Année Universitaire 2016- 2017

Automatisme Logique- 32

Algèbre de Boole

- **Représentation des variables et fonctions logiques**
  - Algébrique (forme littérale) :
    - Équation, proposition, expression
    - Formes technologiques
    - Formes canoniques
  - Graphique :
    - Table de vérité
    - Tableau de Karnaugh
    - Diagramme d'Euler ou de Venn (théorie des ensembles)
  - Temporelle :
    - Chronogramme
  - Symbolique :
    - Logigramme
  - Numérique (écriture condensée)

Pr. K. BENJELLOUN | Année Universitaire 2016- 2017

2. La logique combinatoire

Les systèmes de numération

- **Tout nombre peut s'exprimer sous sa forme polynomiale :**

$$N = \sum_{i=0}^{n} a_i \times b^i$$

- **Dans cette équation polynomiale:**
  - **b = base du système de numérotation**
  - **i = rang ou poids d'un nombre**
  - **a = nombre appartenant à {0,1, ... , (b-1)}**
- **Exemple:**
  - **$(1997)_{10} = 1x10^3 + 9X10^2 + 9x10^1 + 7x10^0$**
  - **Poids du chiffre 1 = 1000**
  - **Rang du chiffre 1 = 3**

2. La logique combinatoire

Les systèmes de numération

Les principales bases

- **Base Décimale (b = 10):**
  - a Č {0,1,2,3,4,5,6,7,8,9}
- **Base Binaire (b = 2)**
  - a Č {0,1}
- **Base Octale (b = 8)**
  - a Č {0,1,2,3,4,5,6,7}
- **Base Hexadécimale (b = 16)**
  - a Č {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}

2. La logique combinatoire

Les systèmes de numération

Changements de base

- **Représentation de nombres décimaux**
  - De la base b à la base décimale
  - De la base décimale à la base b
- **Représentation de nombres binaires**
  - De binaire à octal
  - De octal à binaire
  - De binaire à hexadécimal
  - De hexadécimal à binaire

2. La logique combinatoire

Les systèmes de numération

De la base b à la base décimale (base 10)

- **Ecrire simplement la forme polynomiale, puis calculer.**
- **Exemples:**
  - $(237)_8 = 2x8^2 + 3x8^1 + 7x8^0 = (159)_{10}$
  - $(56A)_{16} = 5x16^2 + 6x16^1 + 10x16^0 = 1386$
  - $(101)_2 = 1x2^2 + 0x2^1 + 1x2^0 = (5)_{10}$

Automatisme Logique- 37

2. La logique combinatoire

Les systèmes de numération

De la base décimale à la base b

- **Deux techniques:**
  - Soustractions successives

    Exemple: $(1386)_{10} = (?)_{16}$

Solution de l'exemple:

- 1386 - 256 = 1130 ; 1130 - 256 = 874; 874 - 256 = 618
- 618 - 256 = 362 ; 362 - 256 = 106 ; 106 - 16 = 90
- 90 - 16 = 74 ; 74 - 16 = 58 ; 58 - 16 = 42
- 42 - 16 = 26 ; 26 - 16 = 10

Donc le nombre commence par un 5, le second nombre est un 6 et le troisième est un 10 ou un A

Solution: $(1386)_{10} = (56A)_{16}$

Pr. K. BENJELLOUN Année Universitaire 2016- 2017

Automatisme Logique- 38

2. La logique combinatoire

Les systèmes de numération

Exemple: $(363)_{10}$ en base 2 ?

Recherche de la puissance 2 juste supérieure = $2^9 = 512$

$363 - 2^8 = 107$	1	MSB
2^7 trop grand	0	
$107 - 2^6 = 43$	1	
$43 - 2^5 = 11$	1	
2^4 trop grand	0	
$11 - 2^3 = 3$	1	
2^2 trop grand	0	
$3 - 2^1 = 1$	1	
1	1	LSB

$(363)_{10} = (101101011)_2$

Pr. K. BENJELLOUN Année Universitaire 2016- 2017

2. La logique combinatoire

Les systèmes de numération

- Divisions successives:

- Exemple: $(1386)_{10} = (?)_{16}$

  Solution de l'exemple:

  - 1386 ÷ 16 = 86 reste 10 (ou A)
  - 86 ÷ 16 = 5 reste 6
  - 5 ÷ 16 = 0 reste 5

  Donc le nombre est $(56A)_{16}$

2. La logique combinatoire

Les systèmes de numération

$(363)_{10}$ en base 2 ?

363	2	181	1
181	2	90	1
90	2	45	0
45	2	22	1
22			
2	2	1	0

$(363)_{10}$ en base 16 ?

363	16	22	11 (B)
22	16	1	6

2. La logique combinatoire

Les systèmes de numération

De la base binaire à la base octale

- **Conversion en groupant des ensembles de** 3 bits**.**
  - Exemple: $(10010110)_2 = (?)_8$
- **Rappel:**
  - 000 = 0 ; 001 = 1 ; 010 = 2 ; 011 = 3
  - 100 = 4 ; 101 = 5 ; 110 = 6 ; 111 = 7
- **Solution de l'exemple:**
  - $(0\underline{10}\ \underline{010}\ \underline{110})_2 = (226)_8$

2. La logique combinatoire

Les systèmes de numération

De la base octale à la base binaire

- **Opération inverse à la précédente**
- **Exemple:** $\mathbf{(3452)_8 = (?)_2}$
- **Solution de l'exemple:**
  - $(3452)_8 = (\underline{011}\ \underline{100}\ \underline{101}\ \underline{010})_2$

2. La logique combinatoire

Les systèmes de numération

De la base binaire à la base hexadécimale

- **Conversion en groupant des ensembles de** 4 bits**.**
- **Exemple:** $(100101101)_2 = (?)_{16}$
- **Solution de l'exemple:**
  - $(000\underline{1}\ \underline{0010}\ \underline{1101})_2 = (12D)_8$

De la base hexadécimale à la base binaire

- **Opération inverse à la précédente**
- **Exemple:** $(3F5B)_{16} = (?)_2$
- **Solution de l'exemple:**
  - $(3F5B)_{16} = (\underline{0011}\ \underline{1111}\ \underline{0101}\ \underline{1011})_2$

2. La logique combinatoire

Les systèmes de numération

Opérations mathématiques en binaires

- **Addition**

La table d'addition :

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 et report de 1

- **Soustraction**

La table de soustraction :

0 - 0 = 0

0 - 1 = 1 et retenue de 1

1 - 0 = 1

1 - 1 = 0

2. La logique combinatoire

Les systèmes de numération

Soustraction (suite)

Complément à 1 :

S'obtient en complémentant le nombre binaire.

Ex.	**A =**	**1 0 1 1 0 1 1 1 0 0 1 0**
Complément à 1 de	**A =**	**0 1 0 0 1 0 0 0 1 1 0 1**

- **Complément à 2 :**

S'obtient en ajoutant 1 au complémentant à 1.

Ex.	**A**	**=**	**1 0 1 1 0 1 1 0 1 0 0 0**
	/A	**=**	**0 1 0 0 1 0 0 1 0 1 1 1**
Complément à 2 de A = /A+1		**=**	**0 1 0 0 1 0 0 1 1 0 0 0**

2. La logique combinatoire

Les systèmes de numération

Soustraction (suite)

Soustraction par complémentation à 2 et addition

Ex.		**1 0 1 1 1 0 1 1 1 0 1**	
	-	**0 0 1 0 1 1 0 0 1 1 0**	**On ajoute des 0s**
		1 0 1 1 1 0 1 1 1 0 1	
	+	**1 1 0 1 0 0 1 1 0 0 1**	**Complément à 1**
	+	**1**	**Complément à 2**
		--	
		1 1 0 0 0 1 1 1 0 1 1 1	**On ignore le report**

2. La logique combinatoire

Les systèmes de numération

Soustraction (suite)

- ❑ Lorsque le bit le plus significatif = 1, le nombre est négatif
- ❑ Le complément à 2 du nombre négatif redonne le même nombre mais avec un signe positif

Codes

- BCD « Binary Coded Decimal »
- Gray ou binaire réfléchi
- ASCII « American Standard Code for Information Interchange »
- Unicode

2. La logique combinatoire

Les systèmes de numération

Code BCD

Décimal Codé Binaire :

Chaque chiffre d'un nombre est codé sur 4 bits

•0	0000	
•1	0001	
•2	0011	
…………		
•10	0001	0000
•11	0001	0001

Ce code simplifie la conversion décimal binaire

2. La logique combinatoire

Les systèmes de numération

Code BCD (Binary coded decimal)

- Souvent utilisé par les machines à calculer.
- Combine les avantages du décimal et du binaire.
- Les chiffres de 0 à 9 suivent le code binaire naturel. Donc les valeurs de A à F ne sont pas utilisées.
- Opérations arithmétiques + complexes.

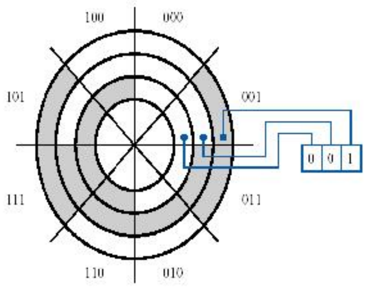
2. La logique combinatoire

Les systèmes de numération

Code Gray

Distance de 1 entre deux mots de code consécutif

0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100



Ce code évite le changement simultané de 2 bits, et donc les états transitoires indésirables.

2. La logique combinatoire

Les systèmes de numération

Code Gray ou binaire réfléchi : code à symétries multiples

Propriétés

distance unité

0 0 0
0 0 1
0 1 1
0 1 0
1 1 0
1 1 1
1 0 1
1 0 0

Cyclique

Monôme de n symboles = n adjacents → multisymétrie du code

Tableau de Karnaugh : codage lignes et colonnes par code Gray

2. La logique combinatoire

Les systèmes de numération

Code ASCII

- **(American Standard Code for International Interchange).**
- **Norme universelle pour la transmission de données.**
  - ASCII normal: 128 caractères sur 7 bits;
  - ASCII étendu: 256 caractères sur 8 bits.
    Norme ISO Latin 1

Code Unicode (ISO 8859-1)

- **Le code ASCII est limité à 256 caractères.**
- **Pour dépasser cette limite, une nouvelle norme sur 16 bits fût créée.**
- **Donc, plus de 65 000 caractères disponibles:**
  - **Japonais, Mandarin, Grec, Russe, Hébreux, Arabe, Coréen, ...**

2. La logique combinatoire

2.3 Types de représentation

- Les fonctions logiques peuvent être représentées de plusieurs façons:
  - Équations logiques
  - Tables de vérités
  - Logigrammes
  - Diagrammes échelle (Ladder)

- Ces représentations seront introduites avec les fonctions de base...

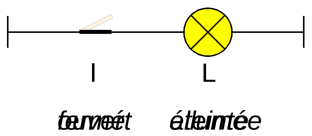
Types de représentation

- Tables de vérités
  - Tables qui énumèrent toutes les combinaisons possibles d'entrées, et les sorties correspondantes.
  - Le nombre de colonnes est la sommes du nombre d'entrée et de sortie
  - Pour "N" entrées, le nombre de lignes est 2^N
- Exemple:

  3 entrées et 1 sorties
  4 colonnes et 8 lignes

Algèbre de Boole

- **La table de vérité**
  - Soit F, une fonction de n variables. La table de vérité de F est un tableau de $n+1$ colonnes et 2^n lignes dans lequel apparaissent toutes les combinaisons d'entrées associées à la valeur correspondante de la fonction.



I	L	
0	0	→ Point faux
1	1	→ Point vrai

Types de représentation

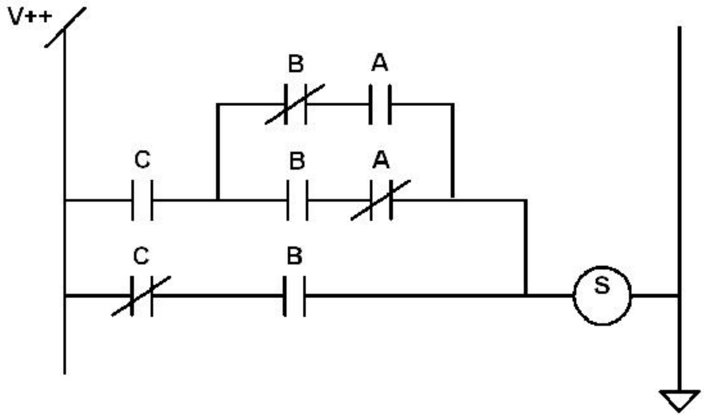
- **Tables de vérités**
  3 entrées et 1 sorties
  4 colonnes et 8 lignes

A	B	C	SORTIE
0	0	0	$\bar{A}\bar{B}\bar{C}$
0	0	1	$\bar{A}\bar{B}C$
0	1	0	$\bar{A}B\bar{C}$
0	1	1	$\bar{A}BC$
1	0	0	$A\bar{B}\bar{C}$
1	0	1	$A\bar{B}C$
1	1	0	$AB\bar{C}$
1	1	1	ABC

- **Chaque ligne est une équation logique**

Types de représentation

- **Diagrammes échelle (Ladder)**



Types de représentation

- **Équations logiques**
  - **Reposent sur 3 opérateurs de base:**
    - **ET, OU, NON**
    - **Toutes les équations logiques sont formées de ces 3 opérateurs**

2. La logique combinatoire

2.4 Définitions

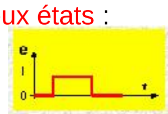
- **Variable logique**: Variable discrète qui peut prendre 2 états associés au caractère vrai ou faux d'un événement
- **Fonction logique**: Ensemble de variables logiques reliées par des opérateurs logiques. Une fonction logique ne peut prendre que 2 valeurs 0 ou 1.
  - Fonction logique combinatoire:
  - Fonction logique séquentielle
- **Table de vérité**: Une table de vérité nous fait connaître la réaction d'un circuit logique (sa valeur de sortie) aux diverses combinaisons de niveaux logiques appliqués aux entrées (2^N)
- **Équation logique**: c'est une expression en fonction des variables logiques et des opérateurs

2. La logique combinatoire

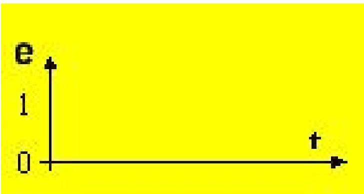
2.4 Définitions

- **Variable logique**:

Une variable logique est une variable qui ne peut avoir que deux états :



1 et 0 en général
ou encore
- Vrai **et** Faux
- Fermé **et** Ouvert
- -5V **et** + 5V (ou bien 0 V et 5V
- **etc...**

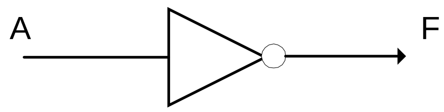


Fonction logique NON

- **En anglais: NOT**
- **Représentation:**
  - $F = \overline{A}$ ou $F = /A$

Table de vérité

Entrée	*Sortie*
A	F
0	1
1	0



Symbole graphique

Algèbre de Boole

- **Opérateurs logiques**
  - opérateurs de bases de l'algèbre de Boole :
    - NON ou PAS (NOT) : fonction complément ou fonction inverse. C'est une fonction f d'une variable x telle que :

$$\boxed{f(x) = \overline{x}}$$

système binaire :

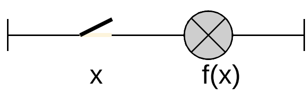
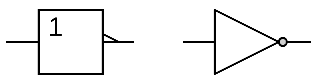


table de vérité :

x	f(x)
0	1
1	0

symbole :



norme CEI norme IEEE

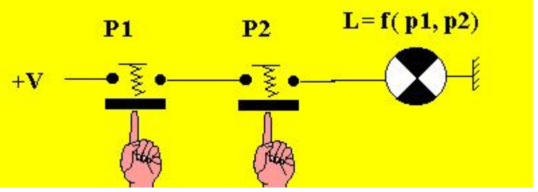
2. La logique combinatoire

2.4 Définitions

Fonction logique:

- Une fonction logique est donc une fonction de variables logiques
- Une fonction logique peut prendre 2 valeurs notées 0 et 1

Exemple: L (état de la lampe)est une fonction logique des variables p1 et p2 liées aux poussoirs



Algèbre de Boole

- ET (AND) : produit logique. C'est une fonction f de plusieurs variables équivalente à l'intersection en théorie des ensembles. **Elle prend la valeur 1 si toutes les variables sont simultanément égales à 1**. Soient x et y, deux variables booléennes, f(x,y) s'écrit:

$$\boxed{f(x,y) = x \cdot y = x \wedge y}$$

système binaire :
Interrupteurs branchés en série

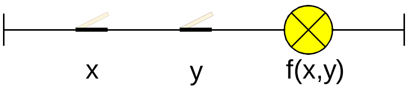
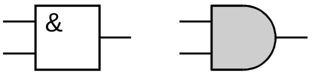


table de vérité :

x	y	f(x,y)
0	0	0
0	1	0
1	0	0
1	1	1

symbole :



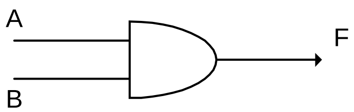
norme CEI norme IEEE

Fonction logique ET

- **En anglais: AND**
- **Représentation:**
  - F = A * B ou A • B ou AB

Table de vérité

Entrée		***Sortie***
B	A	F
0	0	0
0	1	0
1	0	0
1	1	1



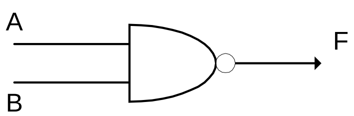
Symbole graphique

Fonction logique NON-ET

- **En anglais: NAND**
- **Représentation:**
  - $F = \overline{A * B}$

Table de vérité

Entrée		***Sortie***
B	A	F
0	0	1
0	1	1
1	0	1
1	1	0



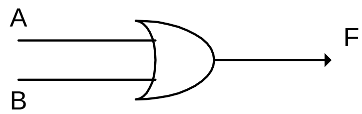
Symbole graphique

Fonction logique OU

- **En anglais: OR**
- **Représentation:**
  - F = A + B

Table de vérité

Entrée		***Sortie***
B	A	F
0	0	0
0	1	1
1	0	1
1	1	1



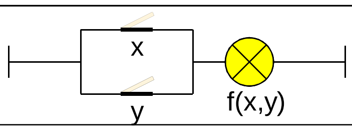
Symbole graphique

Algèbre de Boole

- OU (inclusif) (OR) : somme logique (produel). C'est une fonction f de plusieurs variables équivalente à <u>l'union</u> en théorie des ensembles. **Elle prend la valeur 1 si au moins une variable est égale à 1**. Soient x et y, deux variables booléennes, f(x,y) s'écrit :

$$\boxed{f(x,y) = x + y = x \vee y}$$

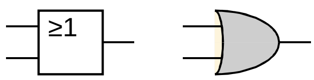
<u>système binaire :</u>
Interrupteurs branchés en parallèle



<u>table de vérité :</u>

x	y	f(x,y)
0	0	0
0	1	1
1	0	1
1	1	1

<u>symbole :</u>



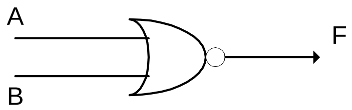
norme CEI norme IEEE

Fonction logique NON-OU

- **En anglais: NOR**
- **Représentation:**
  - $F = \overline{A + B}$

Table de vérité

Entrée		***Sortie***
B	A	F
0	0	1
0	1	0
1	0	0
1	1	0



Symbole graphique

Algèbre de Boole

- opérateurs d'une variable :
  - fonction nulle : f(x) = 0.
  - fonction unité : f(x) = 1.

table de vérité :

x	f(x)
0	0
1	0

table de vérité :

x	f(x)
0	1
1	1

- OUI : fonction identité : f(x) = x.

système binaire :

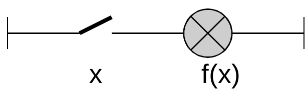
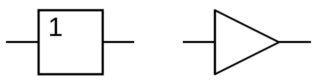


table de vérité :

x	f(x)
0	0
1	1

symbole :



norme CEI norme IEEE

Algèbre de Boole

- opérateurs de deux ou plusieurs variables :
  - OU Exclusif (XOR) : **elle prend la valeur 1 si et seulement si le nombre de variables égales à 1 est impair**. Soient x et y, deux variables booléennes, f(x,y) s'écrit :

$$\boxed{f(x,y) = x \oplus y = \bar{x} \cdot y + x \cdot \bar{y}}$$

système binaire :

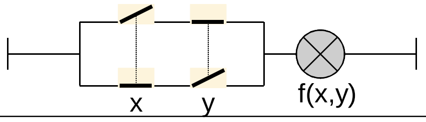
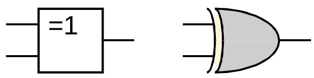


table de vérité :

x	y	f(x,y)
0	0	0
0	1	1
1	0	1
1	1	0

symbole :



norme CEI norme IEEE

Algèbre de Boole

- coïncidence ou identité : **elle prend la valeur 1 si si et seulement si le nombre de variables égales à 1 est pair**. Soient x et y, deux variables booléennes, f(x,y) s'écrit :

$$\boxed{f(x,y) = \overline{x \oplus y} = x \cdot y + \bar{x} \cdot \bar{y}}$$

système binaire :

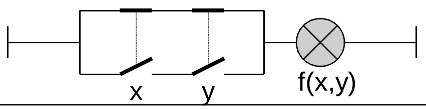
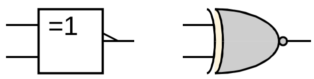


table de vérité :

x	y	x ⊕ y	f(x,y)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

symbole :



norme CEI norme IEEE

Algèbre de Boole

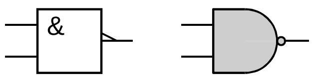
- NON ET (NAND) : **elle prend la valeur 1 si au moins une variable est égale à 0.** C'est un opérateur complet car il permet de réaliser les trois opérateurs de base de l'algèbre de Boole. Soient x et y, deux variables booléennes, f(x,y) s'écrit :

$$f(x,y) = \overline{x \cdot y}$$

table de vérité :

x	y	$x \cdot y$	f(x,y)
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

symbole :



norme CEI — norme IEEE

Algèbre de Boole

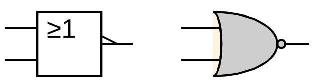
- NON OU (NOR) : **elle prend la valeur 1 si toutes les variables sont simultanément égales à 0.** C'est aussi un opérateur complet. Soient x et y, deux variables booléennes, f(x,y) s'écrit :

$$f(x,y) = \overline{x + y}$$

table de vérité :

x	y	$x + y$	f(x,y)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

symbole :



norme CEI — norme IEEE

Algèbre de Boole

- **Table des fonctions logiques f à 2 variables x et y :**

	(0,0)	(0,1)	(1,0)	(1,1)	f
f0(x,y)	0	0	0	0	0
f1(x,y)	0	0	0	1	$x \cdot y$
f2(x,y)	0	0	1	0	$x \cdot \overline{y}$
f3(x,y)	0	0	1	1	x
f4(x,y)	0	1	0	0	$\overline{x} \cdot y$
f5(x,y)	0	1	0	1	y
f6(x,y)	0	1	1	0	$x \oplus y$
f7(x,y)	0	1	1	1	$x + y$
f8(x,y)	1	0	0	0	$\overline{x + y}$
f9(x,y)	1	0	0	1	$\overline{x \oplus y}$
f10(x,y)	1	0	1	0	$\overline{y}$
f11(x,y)	1	0	1	1	$x + \overline{y}$
f12(x,y)	1	1	0	0	$\overline{x}$
f13(x,y)	1	1	0	1	$\overline{x} + y$
f14(x,y)	1	1	1	0	$\overline{x \cdot y}$
f15(x,y)	1	1	1	1	1

Algèbre de Boole

- **Propriétés et théorèmes**
  - identité (élément neutre) :
    - $A + 0 = A$
    - $A \cdot 1 = A$

A	0	$A + 0$
0	0	0
1	0	1

A	1	$A \cdot 1$
0	1	0
1	1	1

  - involution :
    - $\overline{\overline{A}} = A$

A	$\overline{A}$	$\overline{\overline{A}}$
0	1	0
1	0	1

  - complémentarité :
    - $A + \overline{A} = 1$
    - $A \cdot \overline{A} = 0$

A	$\overline{A}$	$A + \overline{A}$
0	1	1
1	0	1

A	$\overline{A}$	$A \cdot \overline{A}$
0	1	0
1	0	0

Algèbre de Boole

- commutativité :
  - A + B = B + A
  - A · B = B · A

A	B	A · B	B · A
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

- associativité :
  - A + (B + C) = (A + B) + C
  - A · (B · C) = (A · B) · C

A	B	C	B·C	A·(B·C)	A·B	(A·B)·C
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	1	0
1	1	1	1	1	1	1

Algèbre de Boole

- distributivité :
  - ET sur OU : A · (B + C) = (A · B) + (A · C) = AB + AC
  - OU sur ET : A + (BC) = (A + B) · (A + C)

A	B	C	B·C	A+(BC)	A+B	A+C	(A+B)·(A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

Algèbre de Boole

- idempotence (pas d'exposant ou de coefficient) :
  - $A \cdot A = A$
  - $A + A = A$

A	A	A · A
0	0	0
1	1	1

A	A	A + A
0	0	0
1	1	1

- élément absorbant :
  - $A \cdot 0 = 0$
  - $A + 1 = 1$

A	0	A · 0
0	0	0
1	0	0

A	1	A + 1
0	1	1
1	1	1

- absorption :
  - $A \cdot (A + B) = A$
  - $A + (A \cdot B) = A$
  - *Démontrer algébriquement ces deux relations*

A	B	A·B	A+(A·B)
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

Algèbre de Boole

- De Morgan :
  - $\overline{A + B} = \overline{A} \cdot \overline{B}$
  - $\overline{A \cdot B} = \overline{A} + \overline{B}$

A	B	A·B	$\overline{A \cdot B}$	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

- autre identité *à démontrer algébriquement* :
  - $AB + \overline{A}B = B$
  - $(A + B) \cdot (\overline{A} + B) = B$
  - $A + \overline{A}B = A + B$
  - $A \cdot (\overline{A} + B) = AB$
- principe de dualité
  - L'expression duale de toute expression logique (pas équation) s'obtient en permutant les opérateurs ET et OU et les éléments 0 et 1 apparaissant dans l'expression.

Algèbre de Boole

- Exercice :
  - *En utilisant les définitions, propriétés et théorèmes de l'algèbre de Boole développer et simplifier la fonction définie par l'équation suivante :*

$$\boxed{F(a,b,c,d,e) = a\cdot\overline{b \oplus b\cdot c + ce + \overline{de}}}$$

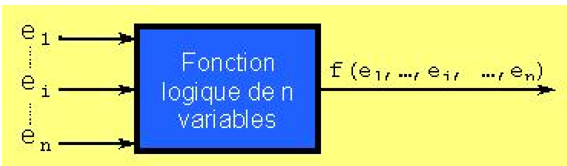
2. La logique combinatoire

2.4 Définitions

Fonction logique combinatoire:

Une fonction logique est dite combinatoire lorsque l'état de la sortie est uniquement définie par la combinaison de l'état des variables logiques d'entrées quelque soit l'instant.

Les fonctions logiques de bases sont (NON, OU, ET)



2. La logique combinatoire

2.4 Définitions

Table de vérité:

Une fonction logique peut être représentée par une table donnant pour toutes les combinaisons des états des variables, l'état correspondante de la fonction.

Elle comporte 2^n lignes (ou n est le nombre de variable), dans l'ordre du binaire naturel.

Cette table est appelée table de vérité

Exemple de table de vérité:

c	b	a	f(a,b,c)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

2. La logique combinatoire

2.4 Définitions

Équation logique:

Une fonction logique peut s'exprimer algébriquement en utilisant l'algèbre de Boole c'est à dire par un groupe de variables reliées par des opérateurs logiques (NON, ET, OU)

On défini tous les états où la fonction est égale à 1 par l'état de toutes les entrées.

Exemple d'équation :

$$F = \overline{c}.b.\overline{a} + \overline{c}.b.a + c.\overline{b}.a + c.b.a$$

2. La logique combinatoire

2.5 Fonctions logiques de base

Représentation des circuits logiques fondamentaux:

2 types de représentations essentiellement:
1. L 'une faisant l 'objet de la norme NFC 03108 de Juillet 1970
2. L 'autre de la norme MIL STD 083 B

c'est cette dernière qui est la plus utilisée

- **Fonction OUI**
- **Fonction NON**
- **Fonction OU**
- **Fonction ET**
- **Fonction NON OU**
- **Fonction NON ET**
- **Fonction INHIBITION**
- **Fonction OU EXCLUSIF**

(Fonction NON OU, Fonction NON ET) Portes universelles

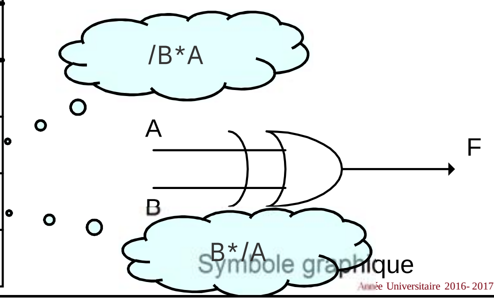
Fonction OU-EXCLUSIF

- **En anglais: XOR**
- **Représentation:**
  - F = ⊕ B

/B*A+B*/A

Table de vérité

Entrée		Sortie
B	A	F
0	0	0
0	1	1
1	0	1
1	1	0



Symbole graphique

Fonction NON OU-EXCLUSIF

- **En anglais: XNOR**
- **Représentation:**
  - F = $\overline{\oplus}$ B

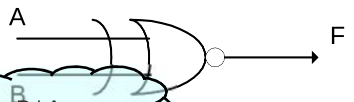
/B*/A+B*A

Table de vérité

Entrée		Sortie
B	A	F
0	0	1
0	1	0
1	0	0
1	1	1

/B*/A

B*A



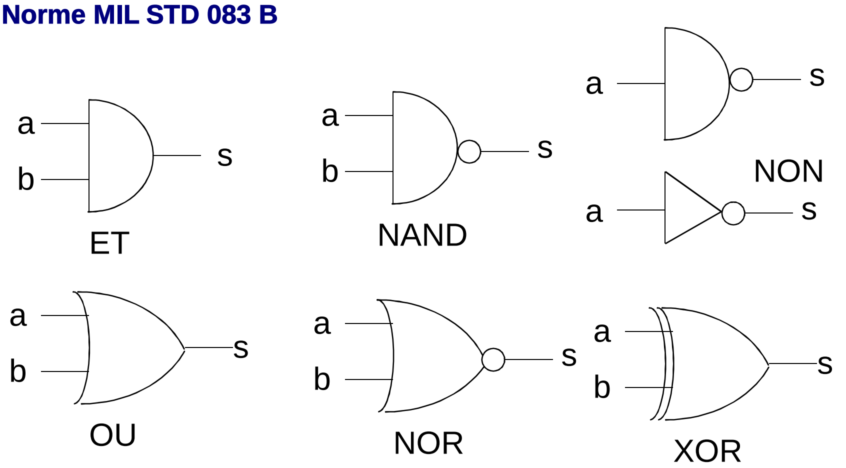
Symbole graphique

2. La logique combinatoire

2.5 Fonctions logiques de base

Représentation graphique :Norme américaine

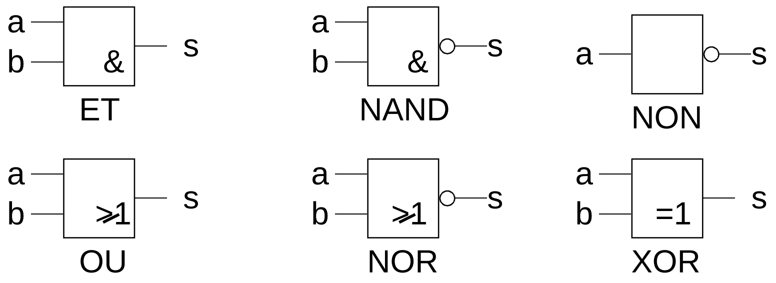
Norme MIL STD 083 B



2. La logique combinatoire

2.5 Fonctions logiques de base

Représentation graphique : Norme française NFC 03108



2. La logique combinatoire

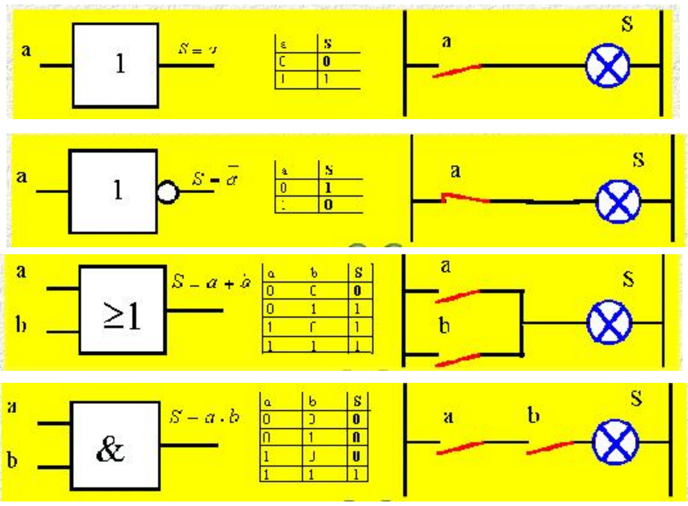
2.5 Fonctions logiques de base

fonction OUI

fonction NON

fonction OU

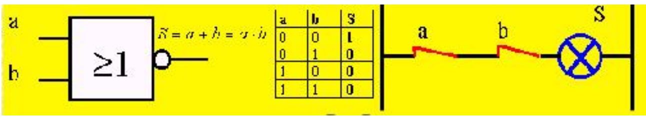
fonction ET



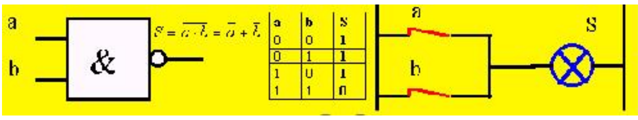
2. La logique combinatoire

2.5 Fonctions logiques de base

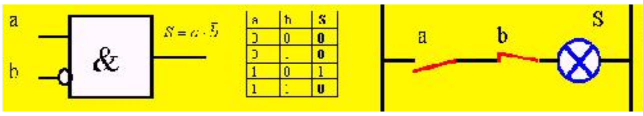
fonction NON OU (NOR)



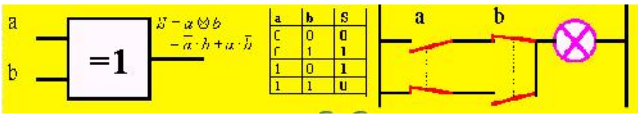
fonction NON ET (NAND)



fct INHIBITION

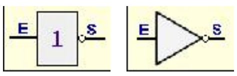
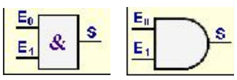
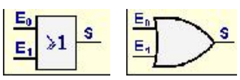


fct OU EXCLUSIF (XOR)



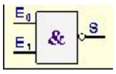
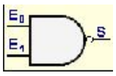
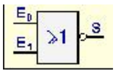
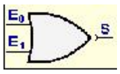
2. La logique combinatoire

2.5 Fonctions logiques de base

Fonction Logique	Symbole	Equation	Table de vérité
NON (NOT)		$\boxed{S = \overline{E}}$ S = /E	E | S: 0 | 1; 1 | 0
ET (AND)		$\boxed{S = E_0 \, . \, E_1}$ $S = E_0 * E_1$	E_1 | E_0 | S: 0 | 0 | 0; 0 | 1 | 0; 1 | 0 | 0; 1 | 1 | 1
OU (OR)		$\boxed{S = E_0 + E_1}$	E_1 | E_0 | S: 0 | 0 | 0; 0 | 1 | 1; 1 | 0 | 1; 1 | 1 | 1

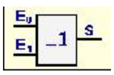
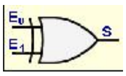
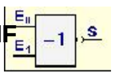
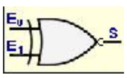
2. La logique combinatoire

2.5 Fonctions logiques de base

Fonction Logique	Symbole	Equation	Table de vérité
NON ET (NAND)	 	$S = \overline{E_0 . E_1}$	E_1 E_0 S 0 0 1 0 1 1 1 0 1 1 1 0
NON OU (NOR)	 	$S = \overline{E_0 + E_1}$	E_1 E_0 S 0 0 1 0 1 0 1 0 0 1 1 0

2. La logique combinatoire

2.5 Fonctions logiques de base

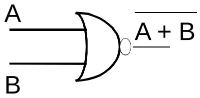
Fonction Logique	Symbole	Equation	Table de vérité
OU EXCLUSIF (EXOR)	 	$S = E_0 \oplus E_1$	E_1 E_0 S 0 0 0 0 1 1 1 0 1 1 1 0
OU NON EXCLUSIF (EXNOR)	 	$S = \overline{E_0 \oplus E_1} = E_0 \odot E_1$	E_1 E_0 S 0 0 1 0 1 0 1 0 0 1 1 1

2. La logique combinatoire

2.5 Fonctions logiques de base

Portes universelles

Grâce aux fonctions NAND et NOR, il est possible de générer toutes les fonctions booléennes.

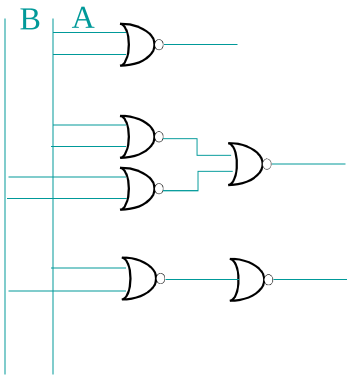


•Ex. Avec NOR

NON **/(A+A) = /A**

ET **/(/A +/B) = //A * //B = A*B**

OU **/(/(A +B)) = A +B**



Fonctions à 2 variables

- **Il existe 16 fonctions logiques possibles avec 2 variables.**
  - Deux variables permettent 4 combinaisons (2^2)
    - 00, 01, 10, 11
  - Ces 4 combinaisons donnent 16 fonctions (2^4)
    - F0, F1, … F15

2. La logique combinatoire

2.5 Fonctions logiques de base

Fonctions de 2 variables

- **Il existe 16 fonctions logiques possibles ayant 2 variables.**

A	B	F0	F1	F2	F3	F4	F5	F6	F7
0	0	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0

A	B	F8	F9	F10	F11	F12	F13	F14	F15
0	0	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1

2. La logique combinatoire

2.5 Fonctions logiques de base

F0 = 0

F1 = /A./B

F2 = /A.B

F3 = /A

F4 = A./B

F5= /B

F6=A⊗B

F7=/(AB)

A	B	F0	F1	F2	F3	F4	F5	F6	F7
0	0	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0

A	B	F8	F9	F10	F11	F12	F13	F14	F15
0	0	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1

$F0 = 0$ $F1 - \bar{A}\bar{B} - \overline{A \quad B}$	$F4 = A\bar{B}$ $F5 - \bar{B}$	$F8 = AB$ $F9 - AB \quad \bar{A}\bar{B} - \overline{A \oplus B}$	$F12 = A$ $F13 - A + \bar{B}$
$F2 - \bar{A}B$	$F6 - \bar{A}B \quad A\bar{B} - A \oplus B$	$F10 - B$	$F14 - A \mid B$
$F3 - \bar{A}$	$F7 = \bar{A} - \bar{B} = \overline{AB}$	$F11 - \bar{A} - B$	$F15 = 1$

Les seize expressions logiques de deux variables d'entrées

Fonctions à 3 variables

- **Il existe 256 fonctions logiques possibles avec 3 variables.**
  - Trois variables permettent 8 combinaisons (2^3)
    - 000, 001, 010, 011, 100, 101, 110, 111
  - Ces 8 combinaisons donnent 256 fonctions (2^8)
    - F0, F1, … F255
  - Pas très convivial !

2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

- circuit électrique

- relais (automatisme)

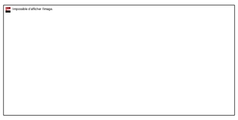
- logigramme (carte de contrôle, circuit intégré,...)

2. La logique combinatoire

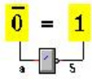
2.6 Réalisations des fonctions logiques :

- circuit électrique

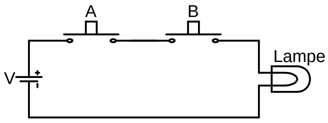
Fonction logique NON



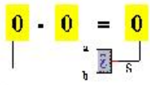
$Lampe = \overline{A}$



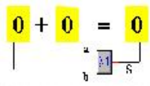
Fonction logique ET

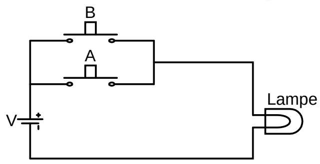


$Lampe = A \bullet B$



Fonction logique OU





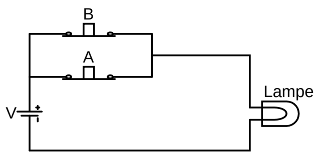
$Lampe = A + B$

2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

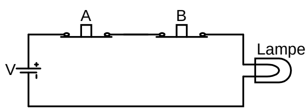
- circuit électrique

Fonction logique NON-ET



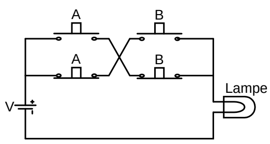
$Lampe = \overline{AB} = \overline{A} + \overline{B}$

Fonction logique NON-OU



$Lampe = \overline{A+B} = \overline{A}\,\overline{B}$

Fonction OU-EXCLUSIF



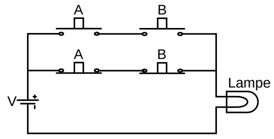
$Lampe = A \oplus B = A.\overline{B} + \overline{A}B$

2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

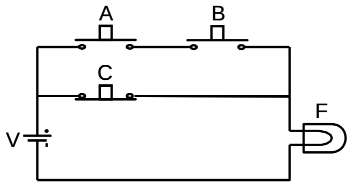
- circuit électrique

Fonction NON OU-EXCLUSIF

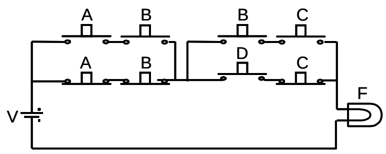


$$Lampe = \overline{A \oplus B} = \overline{A}\,\overline{B} + AB$$

Exercice (1)



Exercice (2)

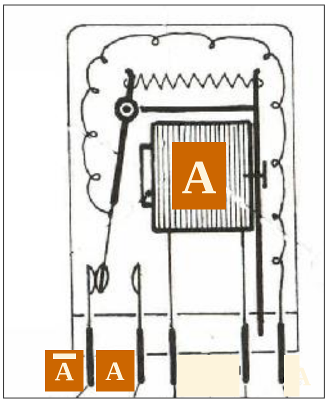


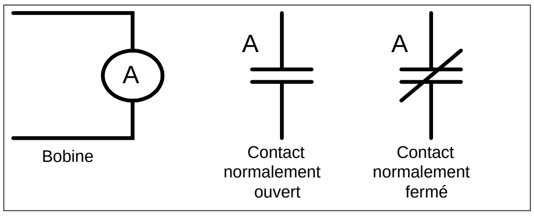
2. La logique combinatoire

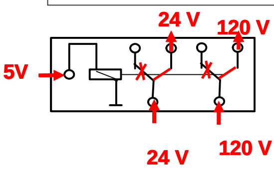
2.6 Réalisations des fonctions logiques :

- relais (automatisme)

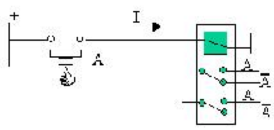
- En automatisation, on utilise les relais pour réaliser des fonctions logiques.
- Le relais est une composante électromécanique.







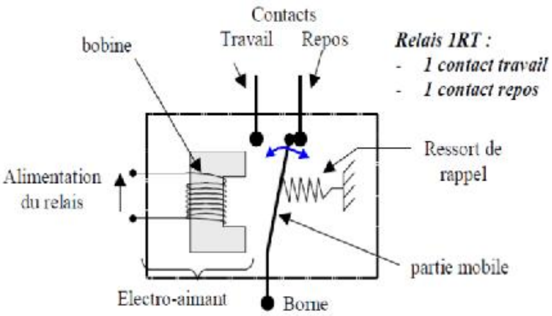
Le relais électrique du Laboratoire



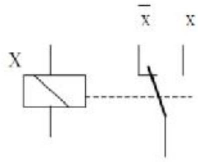
2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

- relais (automatisme)



Représentation schématique du relais 1RT :



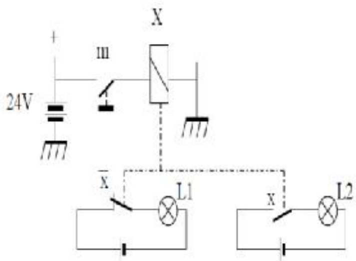
2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

- relais (automatisme)

On représente par X l'état de la bobine du relais et par x et $\bar{x}$ l'état de ses contacts.

- Le contact m est ouvert → X=0: la bobine n'est pas alimentée (hors tension)
  - Le contact travail est ouvert x=0 → L2 est éteinte
  - le contact repos est fermé $\bar{x}$=1 → L1 est allumée
- Le contact m est fermée → X=1: la bobine est alimentée (sous tension)
  - Le contact travail est fermé → x=1 L2 est allumée
  - le contact repos est ouvert → $\bar{x}$=0 L1 est éteinte



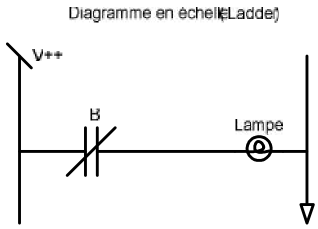
2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

- relais (automatisme)

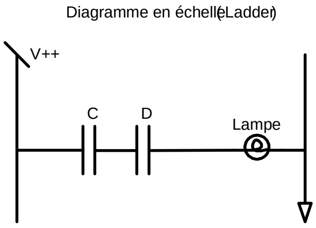
Fonction logique NON

Relais avec un contact normalement fermé



Fonction logique ET

Utilise 2 relais avec des contacts N.O. en séries.



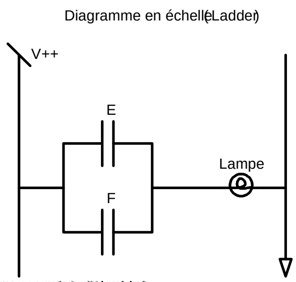
2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

- relais (automatisme)

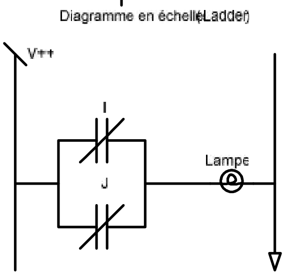
Fonction logique OU

Utilise 2 relais avec des contacts N.O. en parallèles.



Fonction logique NON-ET

Utilise 2 relais avec des contacts N.F. en séries.



2. La logique combinatoire

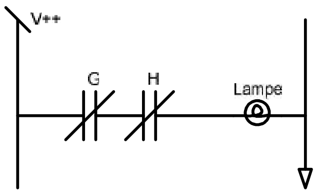
2.6 Réalisations des fonctions logiques :

- relais (automatisme)

Fonction logique NON-OU

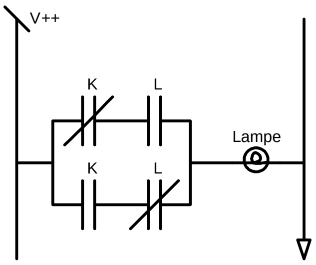
Utilise 2 relais avec des contacts N.F. en parallèles.

Diagramme en échelle (Ladder)



Fonction OU-EXCLUSIF

Diagramme en échelle (Ladder)



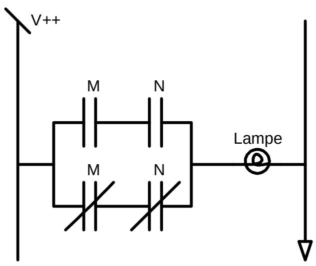
2. La logique combinatoire

2.6 Réalisations des fonctions logiques :

- relais (automatisme)

Fonction NON OU-EXCLUSIF

Diagramme en échelle (Ladder)



Réalisation : exercice

Réaliser (avec des circuits électriques et relais) :

- F = ab + c
- F = (ab + /a/b)(bc + /cd)
- F = (a + b +c)(/a + b/c + c)

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

Règles, postulats et théorèmes, Utiles pour la simplification des équations logiques !

- Toute fonction booléenne d'un nombre quelconque de variables peut s'écrire avec les trois fonctions de base ET, OU et NON
- Fermeture:
  - Si A et B sont des variables Booléennes, alors A+B, A*B sont aussi des variables Booléennes.
- Commutativité
  - A * B = B * A
  - A + B = B + A
- Associativité
  - A + (B + C) = (A + B) + C
  - A * (B * C) = (A * B) * C

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

- **Distributivité**
  - ET sur OU: A(B + C) = AB + AC
  - OU sur ET: A+(B*C) = (A+B)*(A+C)

$2+(3*2) \neq (2+3) * (2+2)$

- **Idempotence**
  - A + A = A
  - A * A = A
- **Complémentarité**
  - $A + \overline{A} = 1$
  - $A * \overline{A} = 0$
  - $\overline{\overline{A}} = A$

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

- **Identités remarquables**
  - 1 + A = 1 et 1 * A = A
  - 0 + A = A et 0 * A = 0
- **Distributivité interne (très utile pour la simplification algébrique des fonctions booléennes).**
  - A + (B + C) = (A + B) + (A + C)
  - A * (B * C) = (A * B) * (A * C)
- **Théorème de De Morgan**
  - $\overline{(A + B)} = \overline{A} * \overline{B}$

  et
  - $\overline{A * B} = \overline{A} + \overline{B}$

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

Fermeture	Si A et B sont des variables booléennes, alors $A+B$, AB sont aussi des variables booléennes
Commutativité	$A+B=B+A$ $A\bullet B=B\bullet A$
Associativité	$A+(B+C)=(A+B)+C$ $A\bullet(B\bullet C)=(A\bullet B)\bullet C$
Distributivité	ET / OU $A(B+C)=AB+AC$ OU / ET $A+(BC)=(A+B)(A+C)$
Idempotence	$A+A=A$ $A\bullet A=A$
Complémentarité	$A+A=1$ $A\bullet\overline{A}=0$
Identités remarquables	$1+A=1$ $1\bullet A=A$ $0+A=A$ $0\bullet A=0$
Distributivité interne	$A+(B+C)=(A+B)+(A+C)$ $A\bullet(B\bullet C)=(A\bullet B)\bullet(A\bullet C)$

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

On retrouve aussi les 8 postulats définis par Boole :

Postulat 1	$0 \bullet 0 = 0$	***Postulat 2***	$1 + 1 = 1$
Postulat 3	$0 \bullet 1 = 1 \bullet 0 = 0$	***Postulat 4***	$0 + 1 = 1 + 0 = 1$
Postulat 5	$1 \bullet 1 = 1$	***Postulat 6***	$0 + 0 = 0$
Postulat 7	$\overline{0} = 1$	***Postulat 8***	$\overline{1} = 0$

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

22 théorèmes dont certains ont été définis par Boole

#1	$0 \bullet A = 0$	***#2***	$1 + A = 1$
#3	$1 \bullet A = A$	***#4***	$0 + A = A$
#5	$A \bullet A = A$	***#6***	$A + A = A$
#7	$A \bullet B = B \bullet A$	***#8***	$A + B = B + A$
#9	$(A \bullet B) \bullet C = A \bullet (B \bullet C)$	***#10***	$(A + B) + C = A + (B + C)$
#11	$A \bullet \overline{A} = 0$	***#12***	$A + \overline{A} = 1$
#13	$A \bullet (B + C) = (A \bullet B) + (A \bullet C)$	***#14***	$A + (B \bullet C) = (A + B)(A + C)$
#15	$\overline{A \bullet B \bullet C \bullet \ldots \bullet Z} = \overline{A} + \overline{B} + \overline{C} + \ldots + \overline{Z}$	***#16***	$\overline{A + B + C + \ldots + Z} = \overline{A}\,\overline{B}\,\overline{C} \ldots \overline{Z}$
#17	$A + AB = A$	***#18***	$A(A + B) = A$
#19	$A + \overline{A}B = A + B$	***#20***	$A(\overline{A} + B) = AB$
#21	$(A + B)(B + C)(\overline{A} + C) = (A + B)(\overline{A} + C)$	***#22***	$AB + BC + \overline{A}C = AB + \overline{A}C$

À partir de tous ces éléments, il est possible de procéder à la simplification des fonctions logiques.

2. La logique combinatoire

2.7 Identités de l'Algèbre de Boole

Exercice

La troisième ligne du tableau nous montre que la distributivité du + donne:

A+B.C =(A+B).(A+C)

- **Montrer que:**

A+B.C.D =(A+B).(A+C).(A+D)

Exemple

- **Trouver l'équation de S.**

Entrées			Sortie
C	B	A	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Exemple

Entrées			Sortie
C	B	A	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

- **Solution:**
  - On construit l'équation de S en écrivant tous les termes donnant S=1.
  - Ainsi, S = 1:
    - si C=0 et B=1 et A=0;
    - ou si C=0 et B=1 et A=1;
    - ou si C=1 et B=0 et A=1;
    - ou si C=1 et B=1 et A=0.

Exemple

Entrées			Sortie
C	B	A	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

- **Solution pour S=1.**
    - si C=0 et B=1 et A=0;
    - ou si C=0 et B=1 et A=1;
    - ou si C=1 et B=0 et A=1;
    - ou si C=1 et B=1 et A=0.
- **On peut donc écrire:**
  - S = /C.B./A + /C.B.A + C./B.A + C.B./A

Exemple

- S = /C.B./A + /C.B.A + C./B.A + C.B./A
- On peut simplifier:

#8 | $A + B = B + A$

¤ S = /C.B./A + C.B./A + /C.B.A + C./B.A

#13 $A \bullet (B + C) = (A \bullet B) + (A \bullet C)$

¤ S = B./A.(/C+C) + /C.B.A + C./B.A

#12 | $A + \bar{A} = 1$

¤ S = B./A.(1) + /C.B.A + C./B.A

#3 | $1 \bullet A = A$

¤ S = B./A + /C.B.A + C./B.A

¤ S = B./A + A.(C ⊕ B) "ou-exclusif"

Exemple

Inspection visuelle ?

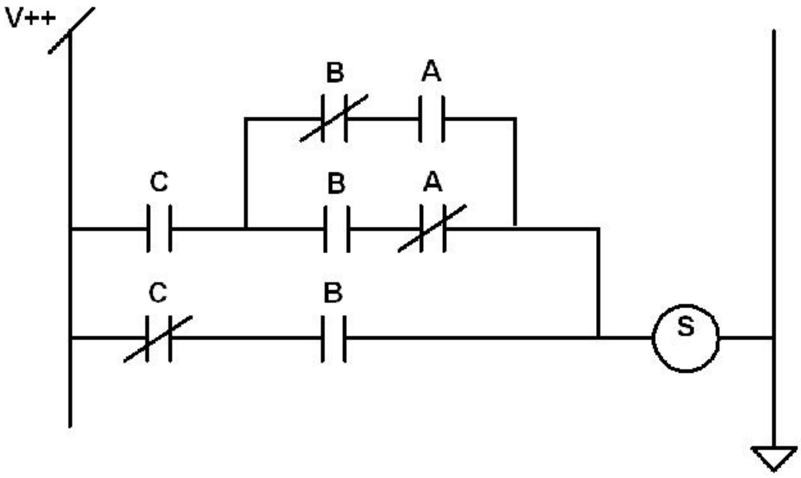
Entrées			Sortie
C	B	A	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

S = /C.B + C./B.A + C.B./A
S = /C.B + C.(A ⊕ B)

S = B./A + /C.B.A + C./B.A
S = B./A + A.(C ⊕ B)

Si nous utilisions des relais...

S = /C.B + C./B.A + C.B./A = C.(/B.A + B./A) + /C.B



2. La logique combinatoire

2.8 Mise en équation d'un circuit

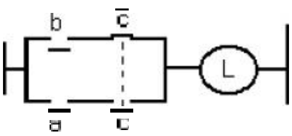
- Étant donné un circuit, il est utile d 'en écrire une équation, même si elle n 'est pas simplifiée,
- En automatismes, la démarche est différente: c'est le concepteur qui décrit la condition de fonctionnement, installe les capteurs adéquats puis traduit la condition de fonctionnement en équation, éventuellement non simplifiée

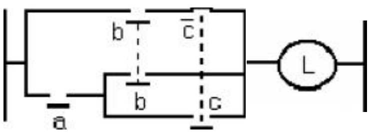
2. La logique combinatoire

2.8 Mise en équation d'un circuit

Exemple

La mise en série de contacts se traduit par un ET logique
La mise en parallèle de contacts se traduit par un OU logique





la fonction L se déduit du fonctionnement même du circuit.

2. La logique combinatoire

- **La simplification est essentielle.**
  - On veut avoir le circuit le plus simple possible...
- **La simplification peut être un processus long si le système est complexe.**
- **Heureusement, il existe des techniques simples pour simplifier.**

2. La logique combinatoire

2.9 Simplification algébrique

- **La traduction de la table de vérité sous forme d'équation logique donne des équations longues et complexes. Il est donc nécessaire de les simplifier.**

Objectifs:

- **Réduire le nombre de variables**
- **Réduire le nombre d'opérateurs (ET, OU, NON)**

pour

- **Simplifier la réalisation**
- **Réduire le nombre de composants**

2. La logique combinatoire

2.9 Simplification algébrique

Principe:

La technique de simplification algébrique exploite l'ensemble des identités de l'algèbre de BOOLE tel que:

- Distributivité
- Commutativité
- De Morgan

Le principe est donc de mettre en forme l'équation pour faire apparaître les identités tels que :

- Complémentation
- Éléments neutres
- Idempotence
- Absorption

2. La logique combinatoire

2.9 Simplification algébrique

Exemple:

$S = \overline{a}.\overline{b}.c + \boxed{\overline{a}.b.c} + \boxed{\overline{a}.b.\overline{c}} + \boxed{\overline{a}.b.c} + \boxed{a.b.\overline{c}}$

distributivité du . $S = b.c\,(a + \overline{a}) + b.\overline{c}\,(a + \overline{a}) + \overline{a}.\overline{b}.c$

complément $S = b.c.(1) + b.\overline{c}\,(1) + \overline{a}.\overline{b}.c$

distributivité du . $S = b.(c + \overline{c}) + \overline{a}.\overline{b}.c = b + \overline{a}.\overline{b}.c$

distributivité du + $S = (b + \overline{a}).(b + \overline{b}).(b + c)$

$S = (b + \overline{a}).(b + c)$

distributivité du + $S = b + \overline{a}.c$

$$S = b + \overline{a}.c$$

2. La logique combinatoire

2.9 Simplification algébrique

Objectif : Fabriquer un système

- à moindre coût
- rapide
- fiable
- peu consommateur

Méthodes : **Algébriques**
Graphiques
Programmables

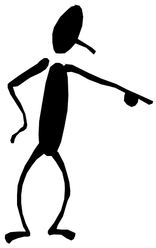
Résultat : on cherche la forme minimale d'une fonction
nombre minimal de monômes/nombre minimal de lettre par monôme

Possibilité de plusieurs formes minimales : formes équivalentes

2. La logique combinatoire

2.9 Simplification algébrique

avertissement



La forme mathématique la plus simple ne correspond pas toujours à la réalisation la plus simple et/ou la plus rapide.

La prise en compte de contraintes technologiques peut imposer une complexification d'écriture de l'expression.

Méthode algébrique toujours possible mais démarche intuitive qui dépend de l'habileté et de l'expérience.

Méthodes de simplification

- **Il est possible d 'obtenir directement une équation sous sa forme simplifiée en utilisant une méthode de simplification graphique.**
- **Méthodes de simplification graphique:**
  - Tables de Karnaugh
  - Tables de Mahoney (non utilisée)

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

La traduction de la table de vérité sous forme d'équation logique donne des équations longues et complexes.

Il est donc nécessaire de les simplifier.

Les tableaux de KARNAUGH **sont une représentation particulière de la table de vérité.**

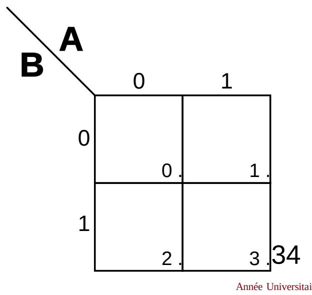
Description : Table de vérité vs <u>Tableau de Karnaugh</u>

1 ligne	**1 case**
n variables	**2^n cases**

Sa conception permet d'obtenir de manière sûre et rapide l'équation la plus simplifiée possible.

Table de Karnaugh

- **Représentation de la table de vérité sous forme graphique.**
- **Nombre de cases = nombre de lignes de la table de vérité.**
  - Multiple de 2^n (1, 2, 4, 8, 16, ...)
    - n = Nombre d 'entrées
  - **Avec n = 2:**
    - Entrées B et A
    - 4 cases



2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

- Le tableau de Karnaugh est représentation géométrique des fonctions logiques utilisant les surfaces (rectangle de Karnaugh). L'intersection de ces rectangle forment des cases. L'ensemble de ces cases forment un tableau. D'où le nom «tableau de Karnaugh »
- Pour une fonction logique f de n variables, le tableau est constitué de la façon suivante:
  - Il comporte 2^n cases: une cases est associée à chaque état d'entrée;
  - Chaque case contient la valeur de la fonction f correspondant à l'état d'entrée associé à cette case.
- Cette représentation est équivalente à celle d'une table de vérité: c'est-à-dire qu'une ligne de la table de vérité correspond à une case de Karnaugh.
- Le tableau de Karnaugh est généralement utilisé pour la simplification des foncions logiques de 3 à 5 variables.

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

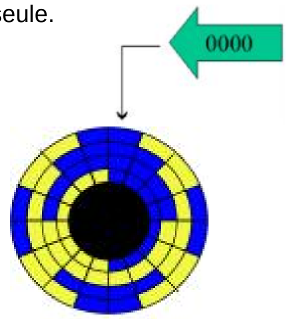
Construire le tableau:

Les cases représentant l'état des variables d'entrée doivent être adjacentes (Une seule variable change d'état -> Code Gray).Les cases adjacentes en ligne et en colonne ne diffèrent que par l'état d'une variable et une seule.

Code Gray

Distance de 1 entre deux mots de code consécutif

0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100



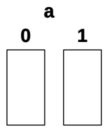
Ce code évite le changement simultané de 2 bits, et donc les états transitoires indésirables.

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

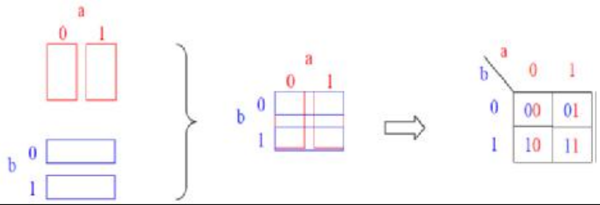
Représentation d'une variable

Soit la variable logique a: Elle peut prendre les valeurs 0 et 1. cette variable est représentée par deux rectangles qui valent 0 ou 1:



Représentation de deux variables

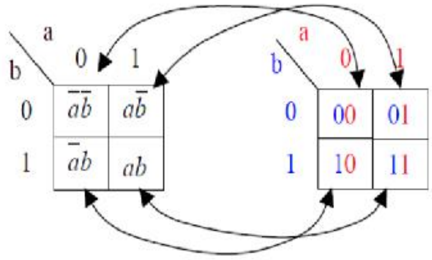
Pour représenter deux variables, on utilise deux fois la représentation d'une variable en superposant les rectangles:



2. La logique combinatoire

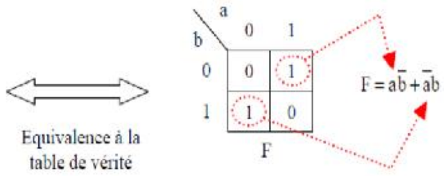
2.9 Simplification par tableau de Karnaugh

A chacune des surfaces élémentaires, on fait correspondre les 4 relations élémentaires pouvant exister dans le cas de deux variables:



La case est alors une des combinaisons de l'état des entrées. Dans cette case, on met alors la valeur que prend la fonction pour la combinaison d'entrée correspondante

a	b	F
0	0	0
0	1	1
1	0	1
1	1	0



2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

Représentation de trois variables

Pour trois variables binaires, il y a 2^3 combinaisons; soit 8 cases

c \ ab	00	01	11	10
0				
1				

Soit la fonction de 3 variables a b et c: $X = a.\overline{b}.\overline{c} + a.\overline{b}.c + \overline{a}.b.c + a.b.c$

c \ ab	00	01	11	10
0	0	2	6	4
1	1	3	7	5

Numérotation des cases sous forme décimale

c \ ab	00	01	11	10
0	0	0	0	1
1	0	1	1	1

Représentation de la fonction X

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

Représentation de quatre variables

Pour quatre variables binaires, il y a 2^4 combinaisons; soit 16 cases

ab \ cd	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

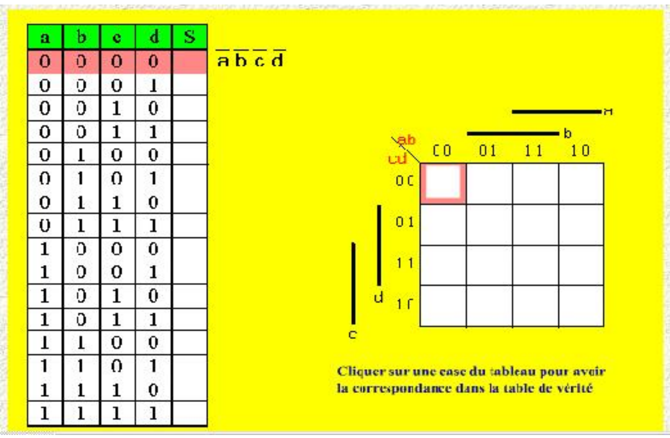
La case 1 » est adjacente à 4 cases: 5,9,12 et 15

Le passage de cette case à l'une des quatre adjacentes ne modifie l'état que d'une seule variable.

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

Exemple: Correspondance Table de vérité / Tableau de Karnaugh

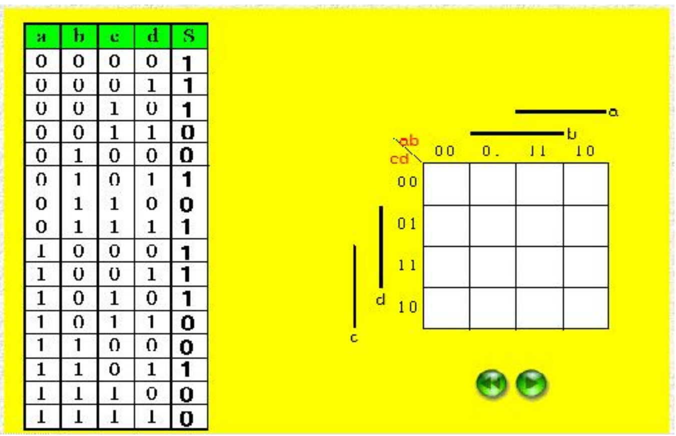


a	b	c	d	S
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

Exemple: Correspondance Table de vérité / Tableau de Karnaugh



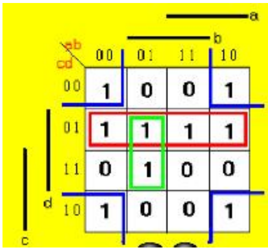
a	b	c	d	S
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

Regrouper:

La méthode de simplification de Karnaugh consiste à rassembler les cases adjacentes contenant des 1 par 2^n cases (2, 4 ou 8...). Les regroupements s'effectuent, symétriquement par rapport aux axes de construction du tableau. Le tableau est sphérique. Les + grands regroupements donnant les équations les + simples. On cherche à avoir le minimum de groupements. Une même case peut intervenir dans plusieurs groupements.

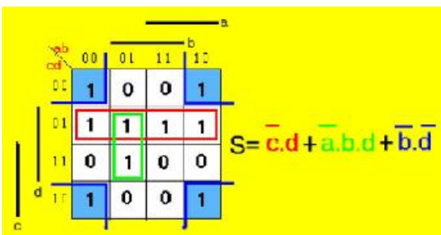


2. La logique combinatoire

2.9 Simplification par tableau de Karnaugh

Écriture de l'équation simplifiée

L'équation de la sortie est une somme de termes, chaque terme étant un regroupement. L'expression logique finale est la réunion des groupements après élimination des variables qui changent d'état. Dans un groupement de deux termes on élimine donc la variable qui change d'état et on conserve le produit des variables qui ne changent pas. A une case correspond un produit de n variables, à 2 cases voisines un produit de n-1 variables, à 4 cases voisines un produit de n-2 variables, etc...



2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Somme de produits + 2 complémentations ⟶ NAND

$$F = a + b.c.d + \bar{b}.\bar{d} = \overline{\overline{a + b.c.d + \bar{b}.\bar{d}}} = \overline{\bar{a}.\overline{b.c.d}.\overline{\bar{b}.\bar{d}}}$$

Produit de sommes + 2 complémentations ⟶ NOR

$$(a+\bar{b}+c).(a+\bar{d}) = \overline{\overline{(a+\bar{b}+c).(a+\bar{d})}} = \overline{\overline{(a+\bar{b}+c)}+\overline{(a+\bar{d})}}$$

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Formes canoniques

Une fonction est sous forme canonique (ou normale) si chaque terme contient toutes les variables. L'écriture sous forme canonique est unique.
Exemple:

$$f(x,y,z) = x.y.z + \bar{x}.\bar{y}.z + \bar{x}.y.z$$

Minterme ou intersection de base

Première forme canonique ou forme normale disjonctive

$$f(x,y,z) = (x+\bar{y}+z).(\bar{x}+y+\bar{z})$$

Maxterme ou réunion de base

Deuxième forme canonique ou forme normale conjonctive

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Formes canoniques

Si la fonction n'est pas sous forme normale

i.e. une des variables (au moins) ne figure pas dans un des termes

$\longrightarrow$ La fonction est sous une forme simplifiée

$f(x,y,z) = xyz + \bar{x}y\bar{z} + xy\bar{z}$ **Première forme canonique**

$= xy(z + \bar{z}) + \bar{x}y\bar{z}$ **Forme simplifiée**

$= y(x + \bar{x}\bar{z})$ **Forme simplifiée**

$= y(x + \bar{z})$ **Forme simplifiée**

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Décomposition de Shannon

Première forme canonique: mise en oeuvre

$$F(a,b) = a.b\,F(1,1) + a.\bar{b}\,F(1,0) + \bar{a}.b\,F(0,1) + \bar{a}.\bar{b}\,F(0,0)$$

Pour chaque a, b la valeur de la fonction F(a,b) dépend du problème

a	b	F	
0	0	0	F(0,0)
0	1	1	F(0,1)
1	0	1	F(1,0)
1	1	0	F(1,1)

$\Longrightarrow F(a,b) = \bar{a}.b + a.\bar{b}$

La première forme canonique ne laisse apparaître que les termes qui valent 1

Il y a 2^N mintermes possibles.

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Deuxième forme canonique: mise en oeuvre

$$F(a,b) = (a + b + F(0,0)).(\bar{a} + b + F(1,0)).$$
$$(a + \bar{b} + F(0,1)).(\bar{a} + \bar{b} + F(1,1))$$

a b	F	
0 0	0	F(0,0)
0 1	1	F(0,1)
1 0	1	F(1,0)
1 1	0	F(1,1)

$$F(a,b) = (a+b).(\bar{a}+\bar{b})$$

Que les termes valant 0

Il y a 2^N maxtermes possibles.

2. La logique combinatoire

Décomposition de Shannon

Nous avions vu la décomposition de Shannon

$$f(x_1, x_2, \ldots, x_n) = x_1 f(0, x_2, \ldots, x_n) + x_1 f(1, x_2, \ldots, x_n)$$

Et son application récursive:

$$f(x_1, x_2, x_3) = \overline{x_1} f(0, x_2, x_3) + x_1 f(1, x_2, x_3)$$

$$f(x_1, x_2, x_3) = \bar{x}_1 \bar{x}_2 f(0, 0, x_3) + \bar{x}_1 x_2 f(0, 1, x_3) + x_1 x_2 f(1, 0, x_3) + x_1 x_2 f(1, 1, x_3)$$

$$\begin{aligned} f(x_1, x_2, x_3) = {} & \bar{x}_1 \bar{x}_2 \bar{x}_3 f(0, 0, 0) + \bar{x}_1 \bar{x}_2 x_3 f(0, 0, 1) + \\ & \bar{x}_1 x_2 \bar{x}_3 f(0, 1, 0) + \bar{x}_1 x_2 x_3 f(0, 1, 1) + \\ & x_1 \bar{x}_2 \bar{x}_3 f(1, 0, 0) + x_1 \bar{x}_2 x_3 f(1, 0, 1) + \\ & x_1 x_2 \bar{x}_3 f(1, 1, 0) + x_1 x_2 x_3 f(1, 1, 1) \end{aligned}$$

2. La logique combinatoire

Décomposition de Shannon (suite)

On a:

$$
\begin{aligned}
f(x_1, x_2, x_3) \quad & \overline{x_1}\,\overline{x_2}\,\overline{x_3}\, f(0,0,0) + \\
& \overline{x_1}\,\overline{x_2}\, x_3\, f(0,0,1) + \\
& \overline{x_1}\, x_2\, \overline{x_3}\, f(0,1,0) + \\
& \overline{x_1}\, x_2\, x_3\, f(0,1,1) + \\
& x_1\, \overline{x_2}\, \overline{x_3}\, f(1,0,0) + \\
& x_1\, \overline{x_2}\, x_3\, f(1,0,1) + \\
& x_1\, x_2\, \overline{x_3}\, f(1,1,0) + \\
& x_1\, x_2\, x_3\, f(1,1,1)
\end{aligned}
$$

On note:

$$
\begin{aligned}
f(x_1, x_2, x_3) \quad & m_0 f(0,0,0) + \\
& m_1 f(0,0,1) + \\
& m_2 f(0,1,0) + \\
& m_3 f(0,1,1) + \\
& m_4 f(1,0,0) + \\
& m_5 f(1,0,1) + \\
& m_6 f(1,1,0) + \\
& m_7 f(1,1,1)
\end{aligned}
$$

2. La logique combinatoire

Décomposition de Shannon (dual)

De la même façon, l'expression duale de la décomposition de Shannon

$$f(x_1, x_2, \ldots, x_n) = (x_1 + f(0, x_2, \ldots, x_n))(x_1 + f(1, x_2, \ldots, x_n))$$

Donne récursivement:

$$f(x_1, x_2, x_3) = (x_1 + f(0, x_2, x_3))(\overline{x_1} + f(1, x_2, x_3))$$

$$
\begin{aligned}
&= (x_1 + x_2 + f(0,0,x_3))(x_1 + x_2 + f(0,1,x_3)) \\
&\quad (\overline{x_1} + x_2 + f(1,0,x_3))(\overline{x_1} + \overline{x_2} + f(1,1,x_3))
\end{aligned}
$$

$$
\begin{aligned}
&= (x_1 + x_2 + x_3 + f(0,0,0))(x_1 + x_2 + \overline{x_3} + f(0,0,1)) \\
&\quad (x_1 + x_2 + x_3 + f(0,1,0))(x_1 + x_2 + x_3 + f(0,1,1)) \\
&\quad (\overline{x_1} + x_2 + x_3 + f(1,0,0))(\overline{x_1} + x_2 + \overline{x_3} + f(1,0,1)) \\
&\quad (\overline{x_1} + \overline{x_2} + x_3 + f(1,1,0))(\overline{x_1} + \overline{x_2} + \overline{x_3} + f(1,1,1))
\end{aligned}
$$

2. La logique combinatoire

Décomposition de Shannon (suite exemple)

On a:

$$f(x_1, x_2, x_3) = (x_1+x_2+x_3+f(0,0,0))\,(x_1+x_2+\overline{x}_3+f(0,0,1))\,(x_1+\overline{x}_2+x_3+f(0,1,0))\,(x_1+\overline{x}_2+\overline{x}_3+f(0,1,1))\,(\overline{x}_1+x_2+x_3+f(1,0,0))\,(\overline{x}_1+x_2+\overline{x}_3+f(1,0,1))\,(\overline{x}_1+\overline{x}_2+x_3+f(1,1,0))\,(\overline{x}_1+\overline{x}_2+\overline{x}_3+f(1,1,1))$$

On note:

$$f(x1, x2, x3) = (M_0+f(0,0,0))\,(M_1+f(0,0,1))\,(M_2+f(0,1,0))\,(M_3+f(0,1,1))\,(M_4+f(1,0,0))\,(M_5+f(1,0,1))\,(M_6+f(1,1,0))\,(M_7+f(1,1,1))$$

2. La logique combinatoire

Forme canonique

Synthèse (exemple)

	A	B	C	S
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

FORME CANONIQUE DISJONCTIVE

Somme de « *mintermes* »

$$S = \overline{A}\,\overline{B}C + \overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}C + ABC$$

Autre forme :

$$S = m_1 + m_3 + m_4 + m_5 + m_7$$

ou bien $S = \sum m(1,3,4,5,7)$

2. La logique combinatoire

Forme canonique (...)

Synthèse (exemple)

	A	B	C	S
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

FORME CANONIQUE CONJONCTIVE

Produit de « *maxterms* »

$$S = \overline{\overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + AB\overline{C}}$$

$$S = \overline{\left(\overline{A}\,\overline{B}\,\overline{C}\right)}\;\overline{\left(\overline{A}B\overline{C}\right)}\;\overline{\left(AB\overline{C}\right)}$$

$$S = \left(A+B+C\right)\left(A+\overline{B}+C\right)\left(\overline{A}+\overline{B}+C\right)$$

Autre forme :

$$S = M_0 M_2 M_6$$

ou bien $S = \prod M(0,2,6)$

2. La logique combinatoire

Minterm, maxterm

Liste des *minterms* et *maxterms* (circuit à trois entrées)

i	A	B	C	m_i	M_i
0	0	0	0	$\overline{A}\,\overline{B}\,\overline{C}$	$\left(A+B+C\right)$
1	0	0	1	$\overline{A}\,\overline{B}C$	$\left(A+B+\overline{C}\right)$
2	0	1	0	ABC	$\left(A+B+C\right)$
3	0	1	1	ABC	$\left(A+B+C\right)$
4	1	0	0	$A\overline{B}\,\overline{C}$	$\left(\overline{A}+B+C\right)$
5	1	0	1	$A\overline{B}C$	$\left(\overline{A}+B+\overline{C}\right)$
6	1	1	0	$AB\overline{C}$	$\left(\overline{A}+\overline{B}+C\right)$
7	1	1	1	ABC	$\left(\overline{A}+\overline{B}+\overline{C}\right)$

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Formes canoniques : Choix

Première forme canonique = expression des 1 de la fonction

Deuxième forme canonique = expression des 0 de la fonction

Les deux formes canoniques sont équivalentes

On choisit celle qui donne le résultat le plus simple

peu de 0 => deuxième forme / peu de 1 => première forme

$$F(a,b) = \bar{a}.b + a.\bar{b}$$

a = msb (Most Significant Bit)
b = lsb (least Significant Bit)
bit = Binary digIT

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Les formes canoniques d'une équation logique

A	B	C	Équation
0	0	0	$\bar{A}\bar{B}\bar{C}$
0	0	1	$\bar{A}\bar{B}C$
0	1	0	$\bar{A}B\bar{C}$
0	1	1	$\bar{A}BC$
1	0	0	$A\bar{B}\bar{C}$
1	0	1	$A\bar{B}C$
1	1	0	$AB\bar{C}$
1	1	1	ABC

- **Forme 1**

$$F = \bar{A}B\bar{C} + AB\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$$

- **Forme 2**

$$F = (A+B+C)(A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+\bar{C})$$

- **Forme 3**

$$F = \overline{\overline{\bar{A}B\bar{C}} \bullet \overline{AB\bar{C}} \bullet \overline{A\bar{B}\bar{C}} \bullet \overline{A\bar{B}C}}$$

- **Forme 4**

$$F = \overline{\overline{(A+B+C)} + \overline{(A+B+\bar{C})} + \overline{(A+\bar{B}+\bar{C})} + \overline{(\bar{A}+\bar{B}+\bar{C})}}$$

2. La logique combinatoire

Réalisation par « NON-ET », « NON-OU »

$$S = A\overline{B} + C$$
$$= \overline{\overline{A\overline{B} + C}}$$
$$= \overline{(\overline{A\overline{B}})\overline{C}}$$

$$\overline{B} = \overline{BB}$$
$$\overline{C} = \overline{CC}$$

$$S = (A+C)(\overline{B}+C)$$
$$= \overline{(A+C)(\overline{B}+C)}$$
$$= \overline{(A+C)} + \overline{(\overline{B}+C)}$$

$$B = B + B$$

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Fonction incomplètement définie

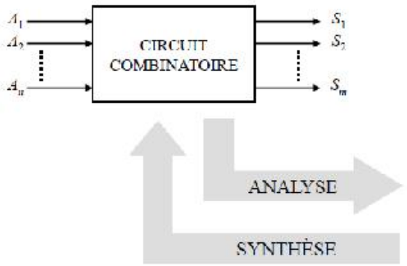
Si une combinaison d'entrée ne peut pas se présenter ou si pour cette combinaison la valeur de la fonction n'est pas importante, on dit que la fonction n'est pas définie en ce point.

$F(a,b,c)$ Z ¤ **(ou x ou -)**

Ce point peut être remplacé par 1 ou 0 en fonction des besoins de simplification.

2. La logique combinatoire

Analyse vs. synthèse



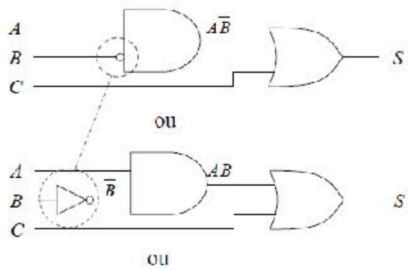
$$S_i = f_i\{A_1, A_2, \ldots, A_n\}$$

A_1	A_2	$\cdots$	A_{n-1}	A_n	S_i
0	0	$\cdots$	0	0	1
0	0	$\cdots$	0	1	1
0	0	$\cdots$	1	0	0
$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$
1	1	$\cdots$	1	1	1

2. La logique combinatoire

Analyse, exemple

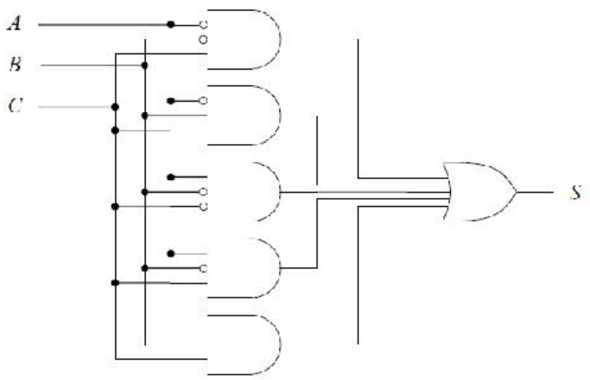
$$S = A\overline{B} + C$$



A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

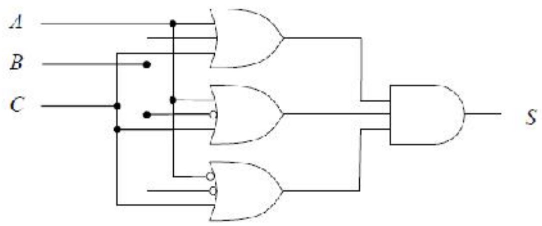
2. La logique combinatoire

$$S = \overline{A}\,\overline{B}C + \overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}C + ABC$$



2. La logique combinatoire

$$S = (A+B+C)(A+\overline{B}+C)(\overline{A}+\overline{B}+C)$$



2. La logique combinatoire

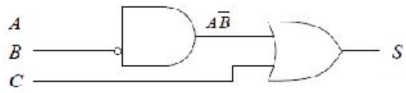
Simplification…

$$S = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + A\bar{B}C + ABC$$

$$\bar{A}C(\bar{B}+B) = \bar{A}C \qquad A\bar{B}(\bar{C}+C) = A\bar{B} \qquad AC(\bar{B}+B) = AC$$

$$(\bar{A}+A)C = C$$

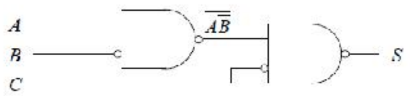
$$S = A\bar{B} + C$$



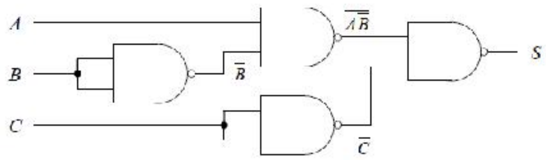
2. La logique combinatoire

Réalisation avec Non-et

$$S = A\bar{B} + C = \overline{\overline{\left(A\bar{B}+C\right)}} = \overline{\left(\overline{A\bar{B}}\right)\bar{C}}$$



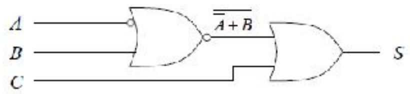
$$\bar{B} = \overline{BB} \qquad \bar{C} = \overline{CC}$$



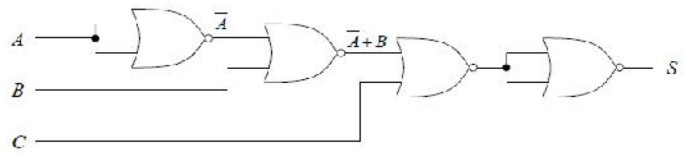
2. La logique combinatoire

Réalisation avec des Non-ou

$S = A\overline{B} + C$ où $A\overline{B} = \left(\overline{\overline{A} + B}\right)$



$\overline{A} = \overline{A + A}$



2. La logique combinatoire

Coût d'un circuit

Lors de la synthèse d'une table de vérité vers un circuit, il est souhaitable d'obtenir la forme d'équation équivalente qui produira un circuit de taille minimale.

Une métrique permet d'évaluer le coût d'un circuit :

Porte logique avec sortie inversée avec N entrées : coût de N+1
Porte logique avec sortie non inversée avec N entrées : coût de N+2

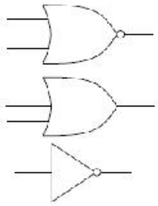
Note : Cette métrique représente un estimé de coût du circuit qui est plus ou moins représentatif de la réalité en fonction de la technologie utilisée pour implémenter le circuit.

2. La logique combinatoire

Coût d'un circuit (...)

Le coût d'une porte avec circuit non inversée est plus grand parce que en technologie CMOS moderne, le circuit équivalent d'une telle porte est une porte inversée suivi d'un inverseur:

Exemple :



Coût
2 + 1 – 3
3 + 2 = 5
2

2. La logique combinatoire

Coût d'un circuit (...)

Exemple de calcul de coût:

On considère que l'on a accès à l'inverse des signaux d'entrée

1. $$S = \overline{A}\,\overline{B}C + \overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}C + ABC$$

5 ET à 3 entrées : 5 (3 + 2) = 25

1 OU à 5 entrées : 5 + 2 – 7

Total – 25 + 7 – 32

2. $$S = A\overline{B} + C$$

1 ET à 2 entrées : 2 + 2 = 4

1 OU à 2 entrées : 2 + 2 – 4

Total = 4 + 4 = 8

3. $$S = \overline{\left(\overline{A} + B\right)\overline{C}}$$

1 OU à 2 entrées : 2 + 2 = 4

1 NON-ET à 2 entrées : 2 + 1 – 3

Total – 4 + 3 – 7

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Exemple (Karnaugh)

Entrées			Sortie
C	B	A	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

TABLE DE VÉRITÉ

C \ BA	00	01	11	10
0	0	0	1	1
1	0	1	1	1

TABLE DE KARNAUGH

Deux termes adjacents par définition et adjacents sur la table de vérité.

2. La logique combinatoire

2.10 Les formes d'écriture d'une fonction logique

Exemple (Karnaugh)

$$S = \overline{C}.B + B.\overline{A} + C.\overline{B}.A$$

C \ BA	00	01	11	10
0	0	0	1	1
1	0	1	0	1

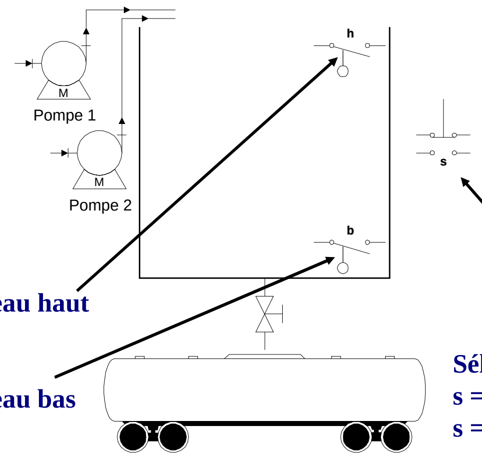
$\overline{C}.B.A + \overline{C}.B.\overline{A} = \overline{C}.B$

$C.\overline{B}.A$

$\overline{C}.B.\overline{A} + C.B.\overline{A} = B.\overline{A}$

2. La logique combinatoire

2.11 Exemple: Contrôle de niveau d'un réservoir



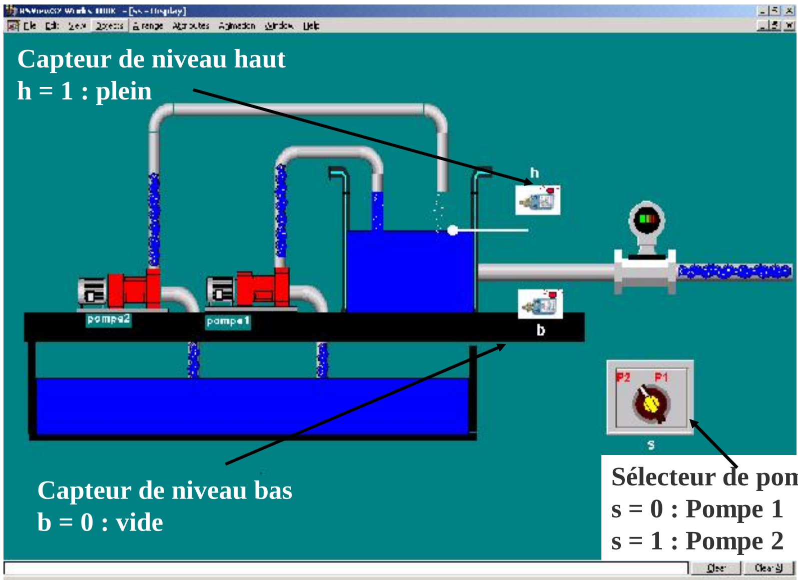
Capteur de niveau haut
h = 1 : plein

Capteur de niveau bas
b = 0 : vide

Sélecteur de pompe
s = 0 : Pompe 1
s = 1 : Pompe 2

2. La logique combinatoire

2.



Sélecteur de pompe
s = 0 : Pompe 1
s = 1 : Pompe 2

2. La logique combinatoire

2.11 Exemple: Contrôle de niveau d'un réservoir

Cahier de charge

Si réservoir plein: Aucune pompe en marche;

Si réservoir vide: Les 2 pompes en marche;

Si réservoir ni vide, ni plein: Faire fonctionner la pompe sélectionnée par le sélecteur « s ».

Capteur de niveau haut : h = 1 ☞ plein

Capteur de niveau bas : b = 0 ☞ vide

Sélecteur de pompe : s = 0 ☞ Pompe 1
s = 1 ☞ Pompe 2

2. La logique combinatoire

2.11 Exemple: Contrôle de niveau d'un réservoir

Solution

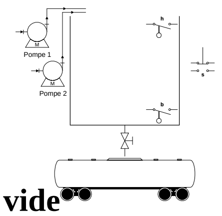


Table de vérité:

Entrées			Sorties	
h	b	s	P_1	P_2
0	0	0	1	1
0	0	1	1	1
0	1	0	1	0
0	1	1	0	1
1	0	0	X	X
1	0	1	X	X
1	1	0	0	0
1	1	1	0	0

Réservoir vide

Réservoir à 1/2

Réservoir plein et vide ?!?

Réservoir plein

2. La logique combinatoire

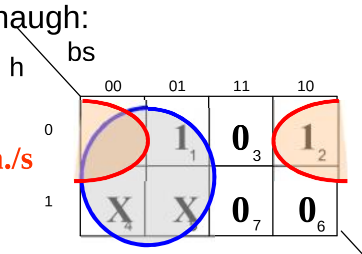
2.11 Exemple: Contrôle de niveau d'un réservoir

Solution

Tables de Karnaugh:

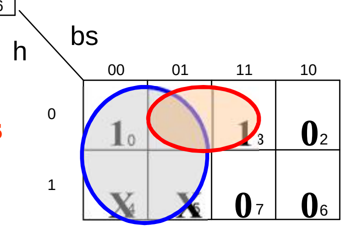
Entrées			Sorties	
h	b	s	P_1	P_2
0	0	0	1	1
0	0	1	1	1
0	1	0	1	0
0	1	1	0	1
1	0	0	X	X
1	0	1	X	X
1	1	0	0	0
1	1	1	0	0

$P_1 = /b + /h./s$



Seul risque: si le capteur b est en panne (b=0) alors que le réservoir est plein → Les deux pompes seront en marche !!!

$P_2 = /b + /h.s$



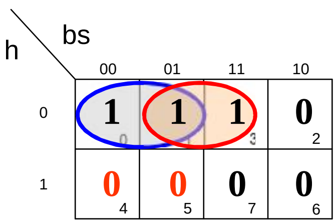
2. La logique combinatoire

2.11 Exemple: Contrôle de niveau d'un réservoir

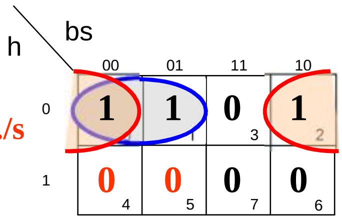
Solution

- **Si on considère les X comme des 0.**

$P_2 = /b./h + /h.s$



$P_1 = /b./h + /h./s$



3. Composants combinatoires

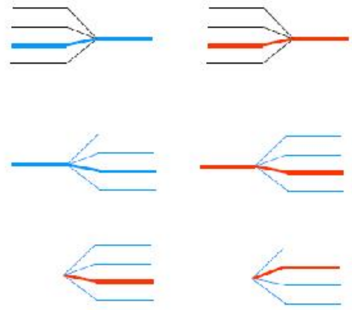
Circuits usuels

Multiplexeur: Sélectionner 1 de 2^n

Démultiplexeur: Passer à 2^n de 1

Décodeur: Passer à 2^n de 0

Encodeur (de priorité): Passer de 2^n à n

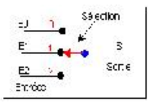


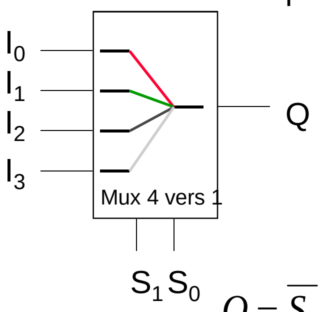
3. Composants combinatoires

Multiplexeur

Le multiplexage consiste à envoyer sur une même ligne de transmission des informations provenant de sources différentes.

Sélection d'une voie parmi 2^N par N bits de comman





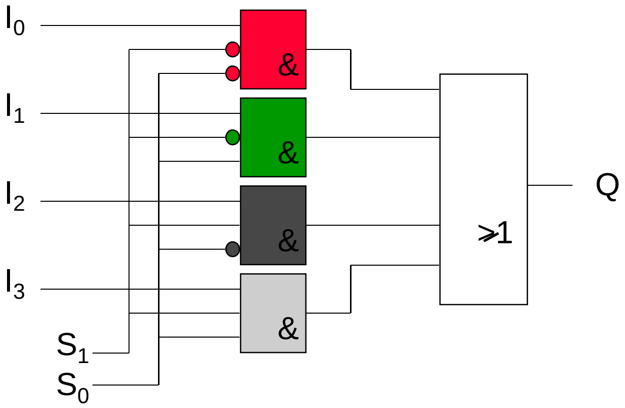
S1	S0	Q
0	0	I0
0	1	I1
1	0	I2
1	1	I3

$$Q = \overline{S_1}.\overline{S_0}.I_0 + \overline{S_1}.S_0.I_1 + S_1.\overline{S_0}.I_2 + S_1.S_0.I_3$$

3. Composants combinatoires

Multiplexeur (constitution)

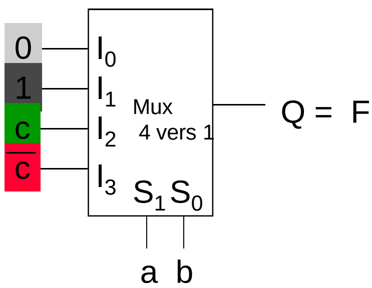
$$Q = \bar{S_1}.\bar{S_0}.I_0 + \bar{S_1}.S_0.I_1 + S_1.\bar{S_0}.I_2 + S_1.S_0.I_3$$



3. Composants combinatoires

Multiplexeur : réalisation de fonctions (2)

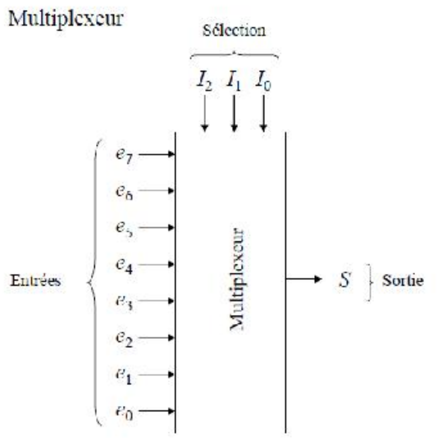
a b c	F	
0 0 0	0	$(ab)_2 = 0$
0 0 1	0	F = 0
0 1 0	1	$(ab)_2 = 1$
0 1 1	1	F = 1
1 0 0	0	$(ab)_2 = 2$
1 0 1	1	F = c
1 1 0	1	$(ab)_2 = 3$
1 1 1	0	F = $\bar{c}$



Toute fonction logique de N variables est réalisable avec un multiplexeur de 2^{N-1} vers 1 et un inverseur

3. Composants combinatoires

Multiplexeur (Mux)

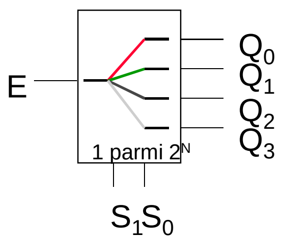


I_2	I_1	I_0	S
0	0	0	e_0
0	0	1	e_1
0	1	0	e_2
0	1	1	e_3
1	0	0	e_4
1	0	1	e_5
1	1	0	e_6
1	1	1	e_7

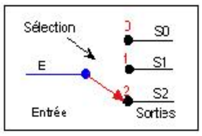
3. Composants combinatoires

Démultiplexeur : 1 parmi 2^n

Le démultiplexage consiste à répartir sur plusieurs lignes à des fins d'exploitations différentes, des informations qui arrivent en série sur une même ligne.



S1	S0	O0	O1	O2	O3
0	0	In	-	-	-
0	1	-	In	-	-
1	0	-	-	In	-
1	1	-	-	-	In

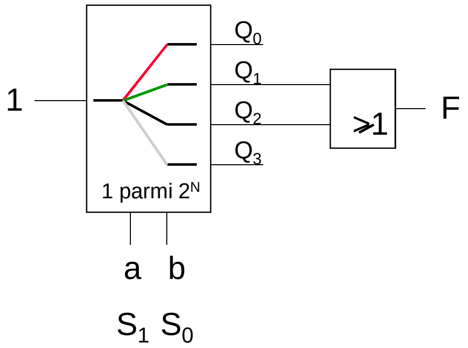


Remarque : E peut ne pas être «disponible»
Sortie sélectionnée = 1 les autres 0
ou Sortie sélectionnée = 0 les autres 1

3. Composants combinatoires

Démultiplexeur: réalisation de fonctions

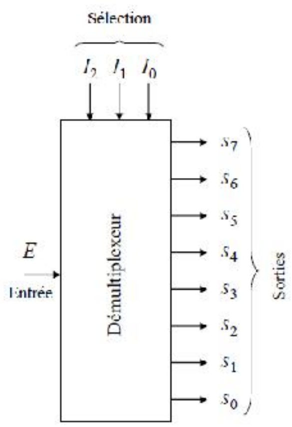
a	b	F	
0	0	0	F(0,0)
0	1	1	F(0,1)
1	0	1	F(1,0)
1	1	0	F(1,1)



3. Composants combinatoires

Démultiplexieur (Démux)

Démultiplexeur (ou convertisseur n bits à 2^n lignes, avec commande d'activation)



I_2	I_1	I_0	s_7	s_6	s_5	s_4	s_3	s_2	s_1	s_0
0	0	0	0	0	0	0	0	0	0	E
0	0	1	0	0	0	0	0	0	E	0
0	1	0	0	0	0	0	0	E	0	0
0	1	1	0	0	0	0	E	0	0	0
1	0	0	0	0	0	E	0	0	0	0
1	0	1	0	0	E	0	0	0	0	0
1	1	0	0	E	0	0	0	0	0	0
1	1	1	E	0	0	0	0	0	0	0

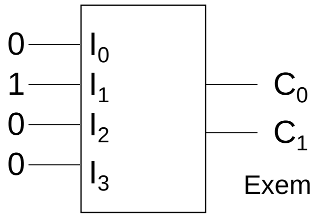
3. Composants combinatoires

Fonction Encodage

Principe: L'encodeur est un système combinatoire ayant pour fonction de retourner l'index d'activation d'une parmi 2^n entrées.L'index d'activation est donné sur n lignes d'adresse. Lorsque plusieurs entrées sont activées, l'encodeur accorde la priorité à l'entrée dont l'index est supérieur.

Faire correspondre un mot code à un symbole

1 entrée parmi N

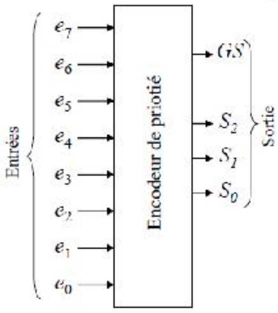


Le code de l'entrée

Exemple : Clavier / Scan code
Caractère / Code ASCII

3. Composants combinatoires

Encodeur (de priorité)



e_7	e_6	e_5	e_4	e_3	e_2	e_1	e_0	S_2	S_1	S_0	GS
1	–	–	–	–	–	–	–	1	1	1	1
0	1	–	–	–	–	–	–	1	1	0	1
0	0	1	–	–	–	–	–	1	0	1	1
0	0	0	1	–	–	–	–	1	0	0	1
0	0	0	0	1	–	–	–	0	1	1	1
0	0	0	0	0	1	–	–	0	1	0	1
0	0	0	0	0	0	1	–	0	0	1	1
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0

3. Composants combinatoires

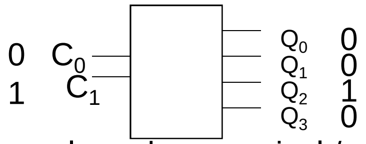
Fonction Décodage

Principe: **Le décodeur est un système combinatoire ayant pour fonction d'activer une des 2^n sorties. La sélection est faite à l'aide de n lignes d'adresse et les sorties sont mutuellement exclusives.**

Remarque : Multiplexeur ↔ Démultiplexeur

Codeur ↔ Décodeur

Décodeur = Démultiplexeur (à E fixe)



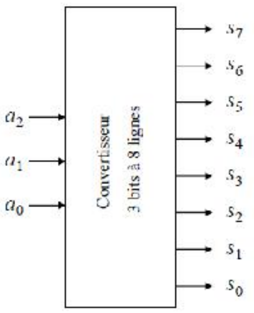
Exemple : adresses pixel / position effective pixel

3. Composants combinatoires

Décodeur

Exemple : Conversion « n bits à 2^n lignes »

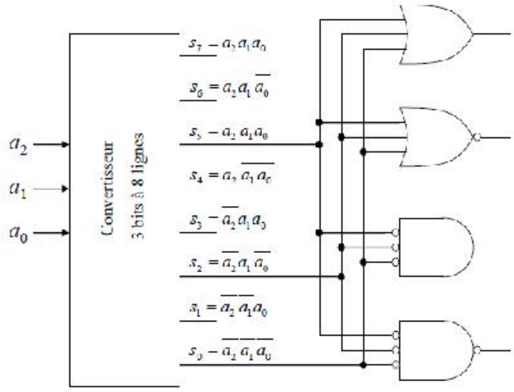
Illustration avec $n = 3$



a_2	a_1	a_0	s_7	s_6	s_5	s_4	s_3	s_2	s_1	s_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

3. Composants combinatoires

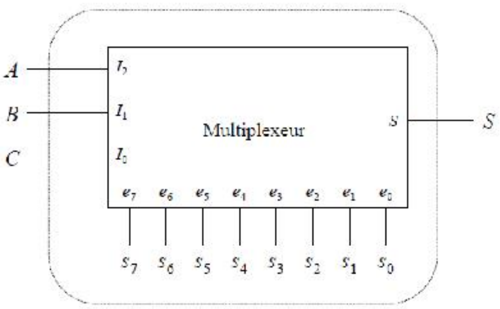
Réaliser une fonction logique



3. Composants combinatoires

Réaliser une fonction logique

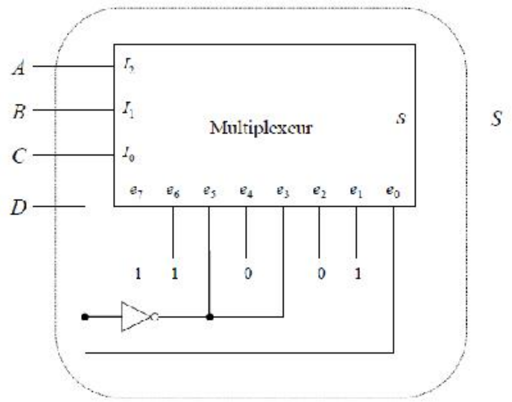
A	B	C	S
0	0	0	s_0
0	0	1	s_1
0	1	0	s_2
0	1	1	s_3
1	0	0	s_4
1	0	1	s_5
1	1	0	s_6
1	1	1	s_7



3. Composants combinatoires

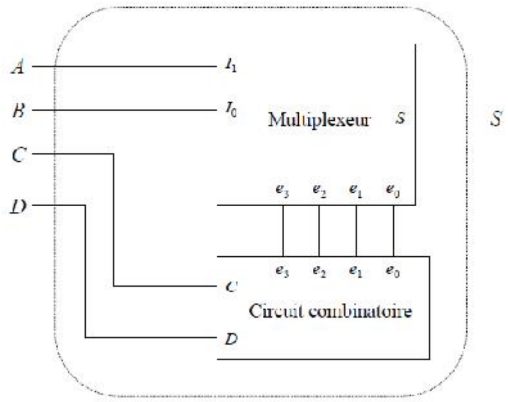
Exemple

A	B	C	D	S	
0	0	0	0	0	D
0	0	0	1	1	
0	0	1	0	1	1
0	0	1	1	1	
0	1	0	0	0	0
0	1	0	1	0	
0	1	1	0	1	$\overline{D}$
0	1	1	1	0	
1	0	0	0	0	0
1	0	0	1	0	
1	0	1	0	1	$\overline{D}$
1	0	1	1	0	
1	1	0	0	1	1
1	1	0	1	1	
1	1	1	0	1	1
1	1	1	1	1	

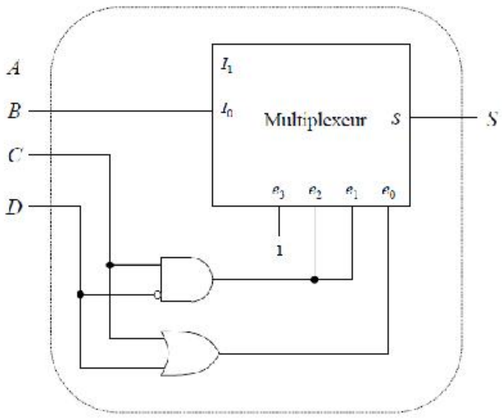


3. Composants combinatoires

Même exemple avec un multiplexeur à 4 entrées



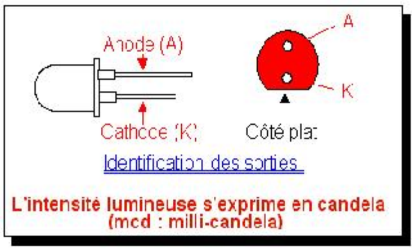
3. Composants combinatoires

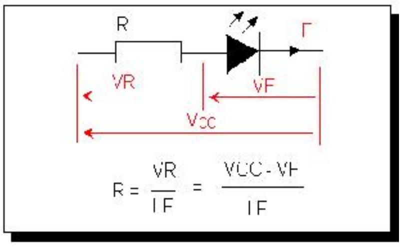


3. Composants combinatoires

Fonction Affichage

Diode électroluminescente DEL (Light Emitting Diode : LED)

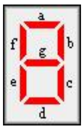


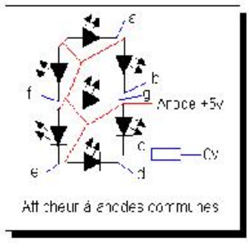


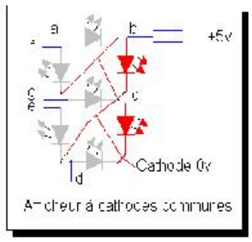
3. Composants combinatoires

Fonction Affichage

Afficheurs sept segments à diodes électroluminescentes



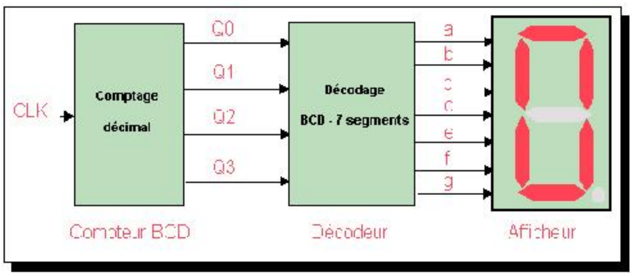




3. Composants combinatoires

Fonction Affichage

Principe de l'affichage: **Pour utiliser un afficheur 7 segments il est nécessaire de disposer d'un décodeur qui traduit le code BCD en code d'allumage des segments de l'afficheur**



3. Composants combinatoires

Décodeur binaire/7 segments

Un décodeur BCD / 7segments permet de commander un afficheur 7 segments (avec sorties de puissance)

Identification des entrées sorties

- **A,B,C et D entrées affectées du mot binaire à afficher en décimal.**
- **BI (Blanking Input) est une entrée supprimant toute affichage sur l'afficheur lorsque cette entrée est placée au niveau bas (Low) : L**
- **a,b,c,d,e,f et g sont les sorties correspondantes à chaque segment de l'afficheur, elles sont actives sur le niveau bas : H**

TTL 74LS48

Boîtier DIL 14 broches
Broche 14 : Vcc (tension alimentation) Broche 7 : GND (masse)

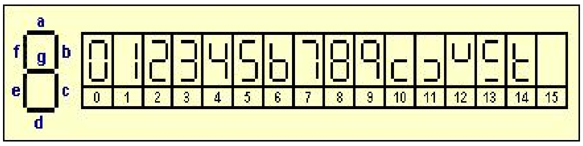
3. Composants combinatoires

Décodeur binaire/7 segments

Table de

Nombre Décimal	Entrées					Sorties						
	[illegible]	[illegible]	[illegible]	[illegible]	DI	a	[illegible]	c	d	[illegible]	f	q
U	U	[illegible]	U	U	1	[illegible]	1	1	1	1	1	U
[illegible]	n	[illegible]	n	I	1	n	1	1	n	[illegible]	n	n
2	0	[illegible]	[illegible]	0	1	[illegible]	1	0	1	1	0	[illegible]
3	0	[illegible]	[illegible]	1	I	[illegible]	I	1	1	[illegible]	0	[illegible]
4	n	1	n	n	1	n	1	1	n	[illegible]	1	[illegible]
5	0	1	0	1	1	[illegible]	[illegible]	1	1	[illegible]	1	[illegible]
6	U	1	[illegible]	U	1	U	[illegible]	1	1	1	1	[illegible]
7	0	1	[illegible]	I	1	[illegible]	1	1	0	[illegible]	0	0
0	1	[illegible]	0	0	1	[illegible]	1	1	1	1	1	[illegible]
9	1	[illegible]	0	1	1	[illegible]	1	1	0	[illegible]	1	[illegible]
10	I	[illegible]	[illegible]	n	1	n	[illegible]	n	I	1	n	[illegible]
1[illegible]	1	[illegible]	[illegible]	1	1	0	[illegible]	1	1	[illegible]	0	[illegible]
12	1	1	U	U	1	U	1	U	U	[illegible]	1	[illegible]
13	1	I	0	1	I	[illegible]	[illegible]	0	1	[illegible]	1	[illegible]
14	1	1	[illegible]	0	1	0	[illegible]	0	1	1	1	[illegible]
16	1	1	[illegible]	1	1	U	[illegible]	U	U	[illegible]	U	U
B	-	-	-	-	[illegible]	U	[illegible]	U	U	[illegible]	U	U

Identification des segments



Automatisme Logique- 201

4. Tables de Karnaugh complexes

Variables inscrites

Ce concept permet de réduire la dimension d'une table de Karnaugh

Illustration :

A	B	C	S
0	0	0	s_0
0	0	1	s_1
0	1	0	s_2
0	1	1	s_3
1	0	0	s_4
1	0	1	s_5
1	1	0	s_6
1	1	1	s_7

A \ BC	00	01	11	10
0	s_0	s_1	s_3	s_2
1	s_4	s_5	s_7	s_6

Pr. K. BENJELLOUN — Année Universitaire 2016- 2017

Automatisme Logique- 202

4. Tables de Karnaugh complexes

Variables inscrites

Ce concept permet de réduire la dimension d'une table de Karnaugh

A	B	C	S
0	0	0	s_0
0	0	1	s_1
0	1	0	s_2
0	1	1	s_3
1	0	0	s_4
1	0	1	s_5
1	1	0	s_6
1	1	1	s_7

$$S = \begin{cases} s_0 & \text{si } C = 0 \\ s_1 & \text{si } C = 1 \end{cases} = \overline{C}s_0 + Cs_1$$

A	B	S
0	0	$\overline{C}s_0 + Cs_1$
0	1	$\overline{C}s_2 + Cs_3$
1	0	$Cs_4 + Cs_5$
1	1	$Cs_6 + Cs_7$

A \ B	0	1
0	$\overline{C}s_0 + Cs_1$	$\overline{C}s_2 + Cs_3$
1	$\overline{C}s_4 + Cs_5$	$\overline{C}s_6 + Cs_7$

Pr. K. BENJELLOUN — Année Universitaire 2016- 2017

4. Tables de Karnaugh complexes

Variables inscrites

Exemple :

A	B	C	S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

A \ BC	00	01	11	10
0	1	1	0	1
1	0	1	0	0

$$S = \overline{A}\,\overline{C} + \overline{B}C$$

4. Tables de Karnaugh complexes

Variables inscrites

A	B	C	S	
0	0	0	1	$S = \begin{cases} 1 \text{ si } C = 0 \\ 1 \text{ si } C = 1 \end{cases} = 1$
0	0	1	1	
0	1	0	1	$S = \begin{cases} 1 \text{ si } C = 0 \\ 0 \text{ si } C = 1 \end{cases} = \overline{C}$
0	1	1	0	
1	0	0	0	$S = \begin{cases} 0 \text{ si } C = 0 \\ 1 \text{ si } C = 1 \end{cases} = C$
1	0	1	1	
1	1	0	0	$S = \begin{cases} 0 \text{ si } C = 0 \\ 0 \text{ si } C = 1 \end{cases} = 0$
1	1	1	0	

A \ B	0	1
0	1	$\overline{C}$
1	C	0

$S = ????$

4. Tables de Karnaugh complexes

Variables inscrites

Exemples de regroupements pour une formulation disjonctive (somme de produits)

$A \backslash B$	0	1
0	0	C
1	0	C

$S = BC$

$A \backslash B$	0	1
0	0	0
1	$\overline{C}$	$\overline{C}$

$S = A\overline{C}$

$A \backslash B$	0	1
0	C	$\overline{C}$
1	C	$\overline{C}$

$S = \overline{B}C + B\overline{C}$

4. Tables de Karnaugh complexes

Variables inscrites

Remarque : « 1 » peut s'exprimer par $C + \overline{C}$

$A \backslash B$	0	1
0	1	$\overline{C}$
1	C	0

$A \backslash B$	0	1
0	$C+\overline{C}$	$\overline{C}$
1	C	0

$S = \overline{B}C + \overline{A}\overline{C}$

Un autre exemple :

$A \backslash B$	0	1
0	1	1
1	C	0

$A \backslash B$	0	1
0	$C+\overline{C}$	1
1	C	0

$S = \overline{B}C + \overline{A}$

4. Tables de Karnaugh complexes

Variables inscrites

Exemples :

$A \backslash B$	0	1
0	1	0
1	C	0

$A \backslash B$	0	1
0	$C+\overline{C}$	0
1	C	0

$$S = \overline{B}C + \overline{A}\,\overline{B}$$

$A \backslash B$	0	1
0	1	1
1	C	1

$$S = C + \overline{A} + B$$

4. Tables de Karnaugh complexes

Variables inscrites

Regroupements pour une formulation conjonctive (produit de sommes)

Remarque : « 0 » peut s'exprimer par $C\,C$

$A \backslash B$	0	1
0	1	$\overline{C}$
1	C	0

$A \backslash B$	0	1
0	1	C
1	C	$C\,\overline{C}$

$$S = (A+C)(B+C)$$

$A \backslash B$	0	1
0	1	0
1	C	0

$A \backslash B$	0	1
0	1	0
1	C	$C\,\overline{C}$

$$S = (\overline{A} + C)\overline{B}$$

4. Tables de Karnaugh complexes

Variables inscrites

A \ B	0	1
0	1	1
1	C	0

A \ B	0	1
0	1	1
1	C	$C \; \overline{C}$

$$S = \left(\overline{A}+C\right)\left(\overline{A}+\overline{B}\right)$$

A \ B	0	1
0	C	$\overline{C}$
1	C	$\overline{C}$

$$S = (B+C)\left(\overline{B}+\overline{C}\right)$$

A \ B	0	1
0	1	1
1	$\overline{C}$	$\overline{C}$

$$S = A + C$$

A \ B	0	1
0	1	C
1	1	C

$$S = B + C$$

4. Tables de Karnaugh complexes

Variables inscrites

Tables de Karnaugh à variables inscrites avec cas facultatifs

A	B	C	S
0	0	0	s_0
0	0	1	s_1
0	1	0	s_2
0	1	1	s_3
1	0	0	s_4
1	0	1	s_5
1	1	0	s_6
1	1	1	s_7

$$S = \overline{C}s_0 + Cs_1$$

où s_0 et s_1 peuvent être 0, 1 ou non spécifiés (cas facultatifs).

Il y a neuf situations possibles : (un cas facultatif est représenté par un tiret)

s_0	s_1	$\overline{C}s_0 + Cs_1$
0	0	0
0	1	C
0	–	0 ou C
1	0	$\overline{C}$
1	1	1
1	–	$\overline{C}$ ou 1
–	0	$\overline{C}$ ou 0
–	1	C ou 1

4. Tables de Karnaugh complexes

Variables inscrites

Exemple :

A	B	C	D	S	
0	0	0	0	0	$S = D$
0	0	0	1	1	
0	0	1	0	1	$S = 1$
0	0	1	1	1	
0	1	0	0	0	$S = 0$
0	1	0	1	0	
0	1	1	0	–	$S = \overline{D}$ ou 0
0	1	1	1	0	
1	0	0	0		$S = -$
1	0	0	1	–	
1	0	1	0	1	$S = \overline{D}$
1	0	1	1	0	
1	1	0	0		$S = -$
1	1	0	1	–	
1	1	1	0	1	$S = 1$
1	1	1	1	1	

A \ BC	00	01	11	10
0	D	1	$\overline{D}$ ou 0	0
1	–	$\overline{D}$	1	–

$$S = C\overline{D} + \overline{A}\,\overline{B}D + AB$$

4. Tables de Karnaugh complexes

Variables inscrites

Exemple :

A	B	C	D	E	S	
0	0	0	0	0	0	$S = E$
0	0	0	0	1	1	
0	0	0	1	0	1	$S = 1$
0	0	0	1	1	1	
0	0	1	0	0	0	$S = 0$
0	0	1	0	1	0	
0	0	1	1	0	0	$S = 0$
0	0	1	1	1	0	
0	1	0	0	0	0	$S = 0$
0	1	0	0	1	0	
0	1	0	1	0	1	$S = 1$
0	1	0	1	1	1	
0	1	1	0	0	1	$S = 1$
0	1	1	0	1	1	
0	1	1	1	0	1	$S = 1$
0	1	1	1	1	1	

A	B	C	D	E	S	
1	0	0	0	0	0	$S = E$
1	0	0	0	1	1	
1	0	0	1	0	0	$S = E$
1	0	0	1	1	1	
1	0	1	0	0	0	$S = 0$
1	0	1	0	1	0	
1	0	1	1	0	0	$S = 0$
1	0	1	1	1	0	
1	1	0	0	0	0	$S = 0$
1	1	0	0	1	0	
1	1	0	1	0	1	$S = \overline{E}$
1	1	0	1	1	0	
1	1	1	0	0	0	$S = 0$
1	1	1	0	1	0	
1	1	1	1	0	1	$S = 1$
1	1	1	1	1	1	

4. Tables de Karnaugh complexes

Variables inscrites

AB \ CD	00	01	11	10
00	E	1	0	0
01	0	1	1	1
11	0	$\overline{E}$	1	0
10	E	E	0	0

$$S = BD\overline{E} + \overline{B}\,\overline{C}E + \overline{A}\,\overline{C}D + BCD + \overline{A}BC$$

4. Les aléas de transition

Aléas Statiques

<u>**Aléa**</u> **: Événement dépendant d'un hasard défavorable.**

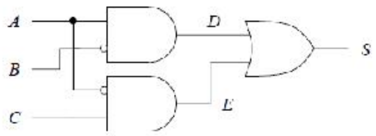
Exemple :

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

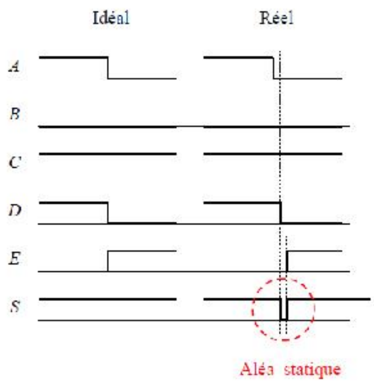
A \ BC	00	01	11	10
0	0	1	1	0
1	1	1	0	0

$$S = A\overline{B} + \overline{A}C$$

4. Les aléas de transition



A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0



4. Les aléas de transition

Solution :

A \ BC	00	01	11	10
0	0	1	1	0
1	1	1	0	0

$$S = A\overline{B} + \overline{A}C + \overline{B}C$$

Terme supplémentaire

En général :

Pour une formulation disjonctive sans aléa statique, tous les « 1 » adjacents doivent être dans un même sous-cube.

4. Les aléas de transition

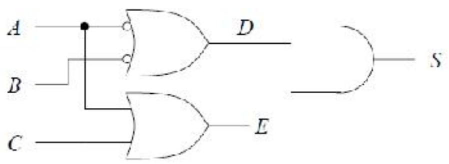
Même exemple selon une formulation conjonctive (produit de sommes) :

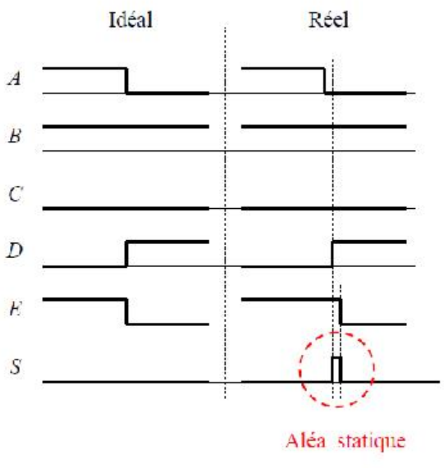
A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

A \ BC	00	01	11	10
0	0	1	1	0
1	1	1	0	0

$$S = \left(\overline{A} + \overline{B}\right)\left(A + C\right)$$

4. Les aléas de transition





A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

4. Les aléas de transition

Solution :

A \ BC	00	01	11	10
0	0	1	1	0
1	1	1	0	0

$$S=\left(\overline{A}+\overline{B}\right)\left(A+C\right)\left(\overline{B}+C\right)$$

Terme supplémentaire

En général :

Pour une formulation conjonctive sans aléa statique, tous les « 0 » adjacents doivent être dans un même sous-cube.